Semantics I-III

Thomas Ede Zimmermann
in collaboration with
Jan Köpping and Dina Voloshina
with a lot of help from Ramona Hiller

Contents

Chapter 0

Background

This is the accompanying material to an introductory course to linguistic semantics – or, more precisely: to *logical semantics*. The general topic, the central questions, as well as some basic considerations and concepts ought be familiar from an introduction to linguistics. The current chapter briefly sums up the most important background assumptions.[1]

0.1 Semantics and Pragmatics

The object of semantics is the literal meaning of linguistic expressions, which include both individual words and, in particular, complex phrases, sentences as well as texts and dialogues. Literal meaning is, as it were, what an expression means all by itself – the meaning it has solely on account of linguistic facts, not on account of its use in a particular context. While, e.g., in Willy Millowitsch's utterance of (1a)[2] the subject may refer to all of mankind as in (1b), former local politician Rüther only talks about himself and those close to him in his utterance of the same sentence, as in (1c):

(1) a. **Wir sind alle kleine Sünderlein.**
[≈ We are all little sinners.]
b. **Die Menschen sind alle kleine Sünderlein.**
[≈ People are all little sinners.]
c. **Die Kölner SPD-Abgeordneten sind alle kleine Sünderlein.**
[≈ The Cologne socialist deputies are all little sinners.]

Who is referring *to whom* by their utterance of the personal pronoun **wir**, i.e., whether (1a) is understood as in (1b), (1c), or whatever, should be clear

[1] More detailed information may be found in the (German) semantics notes to the Frankfurt undergraduate Intro to Linguistics: Link

[2] Willy Millowitsch was a German stage and TV actor. – For those who do not recall the melody his one-time hit: Link

from the circumstances of the utterances – the *context* – and is thus not a question of semantics but of pragmatics, the study of linguistic use and non-literal meaning. On the other hand, anyone who utters (1a) declares himself a sinner – no matter which group he may refer to; for with **wir** [≈ we] speakers *must* refer to themselves – this is demanded by the literal meaning of **wir** (or **we**, for that matter). It is thus a semantic fact that the speakers refer to themselves by (1a).

Unlike the popular actor's utterance of (1a), the local politician's utterance is ironic, if by it he wants to express that the members of his Cologne clique all by no means *little* sinners only. Aspects of irony, too, go beyond the purely literal meaning of sentence (1a) and thus fall into the domain of pragmatics. This also holds of other rhetorical figures of speech and stylistic devices like litotes and metaphor. Among the central non-literal phenomena treated in pragmatics are also the so-called (conversational) *implicatures*; these are inferences beyond the literal meaning that a speaker invites the hearer to draw in view of the circumstances or wording of the utterance. If, e.g., a final report on a political scandal contains sentence (2a), the reader gathers that at least *not all* deputies submitted receipts for fictional donations; for otherwise the report would hardly have left this unmentioned. One thus infers the truth of (2b) from the (written) utterance of (2a):

(2) a. **Mehrere Abgeordnete haben fingierte Spendenquittungen eingereicht.**
[≈ Several deputies submitted fake donation receipts.]
b. **Nicht alle Abgeordneten haben fingierte Spendenquittungen eingereicht.**
[≈ Not all deputies submitted fake donation receipts.]

On the other hand, from the very wording of (2a) one cannot conclude that even a single deputy has a clean record. For (2a) could, e.g., be part of a newspaper article that disinterred the first cases of corruption. In this case the reader should take good care not to conclude (2b) from (2a). And if it later turns out that in fact *all* deputies had submitted fake receipts, (2a) would in no way be refuted. It is only the special circumstances of a comprehensive final report that suggest that (2a) be understood so as to not concern all deputies. As in the case of (1a), the context in which the utterance was made plays a crucial rôle for the inference from (2a) to (2b) – a so-called *scalar implicature*. The exact interaction between literal meaning and background knowledge is important for implicatures like these to arise and thus make a large portion of pragmatics – but are outside the realm of semantics.

Both the parliamentary report and the journalistic investigations are concerned with tax receipts. But (2a) could just as well have been used in connection with receipts for the financial office of parliament (assuming its

existence). Its literal meaning even leaves open whether all deputies mentioned submitted their receipts to the same institution. It may, after all, be that some of them have cheated the tax office and others their parliament; and it may be that (2a) describes these multiple allegations of deceit. Of course, that said deputies submitted their receipts *somewhere or to some institution* is clear by the very wording, the literal meaning of (2a) – but it remains open *where* or *to whom*; this can at best be inferred from the context.

The examples suggest the following demarcation criterion between literal and non-literal meaning:

(3) *Invariance*
Whatever is part of the literal meaning of an expression needs to be *contextually invariant*, i.e., is must not vary across contexts in which the expression is uttered.

It is a contextually invariant part of the meaning of (1a) that the speaker refers to a group (s)he belongs to. Which group the speaker refers to varies from context to context and is thus not part of the literal meaning. It is only part of the literal meaning of (2a) that more than one deputy submitted receipts – to the tax office, to parliament, each to their own institution, etc., this being a matter of context again. And it is also a matter of context whether the utterance is intended and may be understood as including (3b) (or *implicating* (3b), to use the correct pragmatic term). Since the latter inference and the implicit addressee of the receipt submissions vary with context, they are not part of the literal meaning.

The criterion of Invariance turns out to be helpful when it comes to deciding whether a particular aspect needs to be left out of semantic consideration; we will at times use it for this purpose. But the criterion is vague and its application is not always clear. (More about this in an exercise!) Moreover, it is unidirectional. *If* an aspect of meaning varies from context to context, *then* according to this criterion, it is not part of the literal meaning. From this one cannot conclude that all invariant aspects are automatically part of literal meaning; for the criterion says nothing about invariant aspects in general. And there are indeed aspects of meaning that are stable across all contexts without being counted as part of literal meaning. Thus a speaker using the word **Köter** [≈ cur] thereby indicates that she is speaking about (representatives of) a species that she personally dislikes. In this the noun **Köter** [≈ cur] differs from the noun **Hund** [≈ dog]. And still such valuations are usually not counted as part of literal meaning, the reason being theory-internal: if speaker's valuations are separated from literal meaning, the relation between the two aspects can be explained better. We will briefly return to this point in Chapter 8 [which is yet to be written].

One area in which drawing the border between literal and non-literal

meaning may be difficult is the domain of *speech acts*, and particularly of *illocutions* (*types of speech acts*). To begin with, many sentences can only be used in a restricted way:

(4) a. **Du hast noch keine Pizza gegessen.**
[≈ You still haven't had any pizza.]
b. **Wieso isst du keine Pizza?**
[≈ Why don't you eat pizza?]
c. **Iss Pizza!**
[≈ Eat pizza!]

By using (4a) one can make an assertion or claim; with (4b) one may inquire for information; with (4c) one can order or advise someone to consume pizza. Are these possibilities part of the literal meaning of the three sentences? One can also understand (4a) as a demand and (4b) as a piece of advice, after all. Still, even in those cases (4a) and (4b) may also be used assertorically or inquisitively, respectively; demand and advice would then be *indirect* speech acts. In any case, Invariance does not exclude that the use potential of a sentence is part of its literal meaning.

Let us take stock. Semantics is only concerned with the literal meaning of linguistic expressions. In particular, meaning aspects that vary with the utterance context fall into the domain of pragmatics. Further phenomena that are excluded from semantics are the speaker's valuations associated with the use of particular words as well as the range of possible uses of particular sentence types.

0.2 Lexical and logical semantics

In the following the meanings of individual words will only concern us marginally. They are the object of *lexical semantics*. The focus of this course, on the other hand, are the meanings of *complex* expressions – i.e., those that consist of more than one word. They are the object of *logical semantics* (aka *compositional semantics*). We will primarily be concerned with the question of how the meanings of complex expressions emerge from their syntactic structure and the meanings of the words of which they consist. It will turn out that, in general, the answer to this question does not require any detailed knowledge of lexical semantics.

The difference between lexical and logical semantics shows both in the range of phenomena typical for them and in the methods needed to investigate them. To give a flavor of this difference, we will briefly address a few questions of lexical semantics.

Word meanings are more readily studied if they are not regarded in isolation but in relation to each other. As a case in point, a comparison of the words **Handwerker** [≈ craftsman] and **Maurer** [≈ bricklayer] reveals that

the latter has a more specific meaning than the former, whereas both are more general than **Polier** [≈ site foreman], though less special than **Person** [≈ person]. In this connection, the relation between more special and more general is to be understood such that the more general designation applies to everything to which the more special one applies, but not *vice versa*: if someone is rightly described as **Polier** [≈ site foreman], one may also describe him by **Maurer** [≈ bricklayer]. In lexical semantics, this relation, which rests on the (literal) meanings of the words so related, is called *hyponymy*: **Polier** [≈ site foreman] is a hyponym of **Maurer** [≈ bricklayer], which in turn is a hyponym of **Handwerker** [≈ craftsman], which in turn is a hyponym of **Person** [≈ person]. Hyponymy is a *sense relation*, a relation between expressions that holds solely in view of the literal meanings of these expressions. Further examples for sense relations [and words that stand in them] are: *synonymy*, which holds between words that have the same literal meaning [**obwohl** [≈ though] : **obgleich** [≈ although]]; *converse*, which holds between expressions that express relations in opposite directions [**Lehrer** [≈ teacher]: **Schüler** [≈ pupil]]; *incompatibility*, which holds between nouns that can never relate to the same object [**Gedanke** [≈ thought] : **Buch**[≈ book]].

Sense relations do not only hold between words. Complex expressions, too, can be synonyms (5a), hyponyms (5b), converses (5c) of one another or incompatible with each other (5d):

(5) a. **lila Apfelsine** : **violette Orange**
[≈ purple orange] : [≈ purple orange]
b. **kleiner grüner Kaktus** : **grüner Kaktus**
[≈ little green cactus] : [≈ green cactus]
c. **zwei Jahre älter** : **zwei Jahre jünger**
[≈ two years older] : [≈ two years younger]
d. **nur sonntags** : **nur werktags und samstags**
[≈ only on Sundays]: [≈ on only workdays and Saturdays]

The examples show that the sense relations between complex expressions do not merely result from those between the words involved. The synonymy in (5a) does not come about because all words of the left expression are synonymous with those of the right expression; for even though **lila** [≈ purple] and **violett** [≈ purple] are synonyms, just like **Orange** [≈ orange] and **Apfelsine** [≈ orange][3] but not **lila** [≈ purple] and **Apfelsine** [≈ orange]. But then

[3]This case is somewhat more complicated in that probably practically no one actively uses **lila** [≈ purple] and **violett** [≈ purple]. (In my experience, **Orange** [≈ orange] and **Apfelsine** [≈ orange] are different in this respect.) But normally speakers of German *understand* both adjectives, at the same time assuming a difference between them though: those who use **lila** [≈ purple] may, e.g., think that **violett** [≈ purple] refers to a brighter color. However, empirical investigations suggest that both words are used to refer to the same color by speakers who actively use them.

the words in (5a) do stand in a pairwise synonymy relation. The hyponymy in (5b) is different in this respect. For the only sense relation of note that holds between the individual words, is again synonymy – as it trivially holds between the two occurrences of **grüner** [≈ green] – but precisely this synonymy does not carry over to the entire expressions; adding the adjective **klein** [≈ little] turns the synonymy into a hyponymy. In the case of (5c), on the other hand, the result of modifying each of the comparatives **älter** [≈ older] and **jünger** [≈ younger] by **zwei Jahre** [≈ two years] preserves the relation of converse between them. Finally in (5d), the incompatibility is obviously essentially created by the logical words **nur** [≈ only] and **und** [≈ and]; for **sonntags** [≈ on Sundays] on its own neither excludes **werktags** [on workdays] nor **samstags** [≈ on Saturdays].

So the sense relations between complex expressions do not directly derive from those between the words involved. Logical semantics has developed methods to account for sense relations like those in in (5) as results of an interaction of word meaning and syntactic constructions. As a result, or a by-product, as it were, a natural classification of lexical sense relations emerges. We will return to this at times during this course. At this point it only needs to be noted that the representation of sense relations can be transferred from the non-lexical to the lexical domain, but not vice-versa.

Another phenomenon frequently encountered in the lexical domain is ambiguity, which occurs when single words or word forms carry more than one meaning. The form **weiß** [≈ knows/white] may be a verb form or the predicative form[4] of an adjective. All forms of the noun **Ball** [≈ ball] (**Balls**, **Bällen** etc.) can refer to spherical play utensils or to dance festivities. In these case it is obvious that we are dealing with two different words that happen to be pronounced and written in exactly the same way. In order to distinguish them from one another, one may, for instance, use subscripts (**Ball**$_1$, **weiß**$_{\text{3.ps.sg.ind.pres.of }\textbf{wissen}}$ etc.). Each of these *disambiguated* words has its own meaning specified in the lexicon.

The cases of lexical ambiguity just mentioned were all *homonyms*, i.e., words or word forms that have several unrelated – or not obviously related – meanings. Not all ambiguous words are homonyms though. The phenomenon of *polysemy* is encountered at least as frequently: ambiguous word forms where it seems that one meaning is derived from the other one or where both seem to have a common core:

- The noun **Glas** [≈ glass] can refer to a material as well as to a type of drinking container, but then there is an obvious connection: typically

[4]The *predicative* form is that form that is used in sentences of the form 'x **ist** ADJECTIVE' – as opposed to an *attributive* form, which serves to modify a noun (**weißes Tuch** [≈ white cloth]).Unlike, e.g., French ones, German adjectives only have one predicative form but usually several attributive ones. (The adjective **lila** [≈ purple] is an exception to this rule.)

containers referred to as **Glas** [≈ glass] are made of the material called **Glas** [≈ glass]. Despite this tight connection we are dealing with two distinct words (cf. the exercise).

- The adjective **kurz** [≈ short] can relate to spatial distances as well as to temporal intervals, and again there is at least this plausible connection: a short interval is one that can be represented in the spatial dimension by a short distance.

- The adjective **scharf** [sharp/spicy] can be used as the opposite of (or antonym to[5]) either **mild** [≈ mild] or **stumpf** [≈ blunt]. Again there seems to be a connection even it is not so easy to pin down.

Calling objects made of some material by the name of the material is quite common: **Gips** [≈ plaster], **Papier** [≈ paper], **Leder** [leather/chammy] etc.; in lexical semantics this is called *metonymic polysemy*[6] The connection between the readings of **kurz**, however, is *metaphoric*: space serves as a picture of time. This type of frozen metaphor[7] is extremely frequent, not only in German [or English, for that matter]. Thus, e.g., many spatial prepositions (**in** [≈ in|, **vor** [≈ before], **zwischen** [≈ between], ...) can also be used in a temporal sense. But there are more metaphors that underly lexical ambiguity: **Untergang** [≈ sinking/decline], **Kreuzung** [crossing/intersection] **Schmalz** [lard/kitsch]. The third of the above examples rests on a transposition of the tactile sense to taste and thus illustrates *synesthetic polysemy*, metaphor across dimensions of perception, as it can also be found in **hell** [bright/high and clear] and **rau** [rough/hoarse].

Apart from the kinds of polysemy mentioned here there are more possible connections between the readings of single words. This variety presents a major problem to lexical semantics. In fact neither a comprehensive classification of types of polysemy nor a full theory of its occurrence have been developed so far. It still seems clear that it is not a totally random phenomenon; rather, there are – cross-linguistic – regularities in the diversification of lexical meanings. Some of these regularities – especially in the area of metonymic polysemy – are pretty well investigated. But a general theory of polysemy still needs to be developed. Not even a clear-cut demarcation from homonymy is known: is the ambiguity of **treffen** [≈ meet/hit] illustrated in (6a) and (6b) a case of polysemy? This can hardly be decided in the absence of a clear criterion.

[5] Antonymy is the sense relation that holds between degree adjectives whose comparatives are converses of each other. Somewhat simplifyingly, antonyms always refer to the opposite poles of a scale: **lang** [≈ long] : **kurz** [≈ short]; **groß** [≈ big] : **klein** [≈ small]; **heiß** [≈ hot]: **kalt** [≈ cold]; etc.

[6] The term covers all kinds of polysemy where there is a factual connection between the two readings – including the readings of **Schule** [≈ school] as a building or an institution.

[7] A *frozen* metaphor is one that has become part of common linguistic usage.

(6) a. **Der Gewährsmann trifft einen Beamten am Schalter.**
[≈ The informant is meeting a civil servant at the counter.]
b. **Die Gewehrkugel trifft einen Bekannten in der Schulter.**
[≈ The bullet hits an acquaintance at the shoulder.]

We will not go into these problems within this course; and whenever dealing with polysemous words, we will treat them like homonyms, only teasing apart the distinct readings (e.g., by subscripts) without making a connection between them.

0.3 Semantics and Syntax

Structural Ambiguity

Ambiguity not only exists in the lexicon. Complex expressions, too, may be ambiguous. For a start, a complex expression may contain one or more ambiguous words:

(7) **Ein Wechsel der Bank bewirkt keinen Unterschied im Gehalt.**

[≈ A change/cheque of the bank/bench does not make a difference in salary/content.]

Three of the four nouns in (7) are ambiguous. So the sentence has at least $2^3 = 8$ readings, some of which do not make much sense though. Apart from this inheritance from lexical to complex expressions, there is a much more interesting type of ambiguity – viz. *structural ambiguity*, which ensues when the same (not necessarily ambiguous) word forms combine into the same sequence of words in different ways:

(8) **Der Bauer schlug einen Esel mit einer Rute.**
[≈ The farmer beat a donkey with a cane.]

All the individual words are unambiguous,[8] (8) is ambiguous. The following paraphrases bring this out:

(9) a. **Mit einer Rute schlug der Bauer einen Esel.**
[≈ Using a cane, the farmer beat a donkey.]
b. **Einen Esel mit einer Rute schlug der Bauer.**
[≈ A donkey with a cane was beaten by the farmer.]

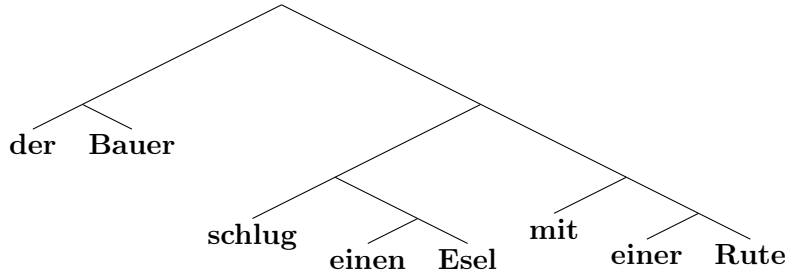
The second reading is less obvious – perhaps because without a particular

[8] More precisely, the lexical ambiguities in (8) are irrelevant in that they can be discarded for syntactic reasons: the noun **Bauer** in the sense of *birdcage* does not occur in (8), because it would have called for the neutral article **das** in lieu of **der**. Something analogous holds of the article **der**, which is ambiguous with respect to case and gender.

context, one does not easily imagine donkeys with canes. But there is no doubt that (8) may be construed in the sense of (9b). This in itself does not prove that we are dealing with a true case of ambiguity, though; it is conceivable, after all, that (8) is indeterminate as to who has the cane. However, a counting test clears this up. For if (8) in the sense (9a) is true of two different occasions, when may infer (8) ... **und zwar zweimal** [≈ and even twice]; the same is true if the farmer twice beat a be-caned donkey (and not necessarily the same one). But if he first uses a cane to beat a caneless donkey and then uses his hand to beat a donkey that does have a cane, the corresponding continuation would not be justified. So one cannot add the occasions at which (9) is true in the sense of (9a) to those at which (8) is true in the sense of (9b).[9]

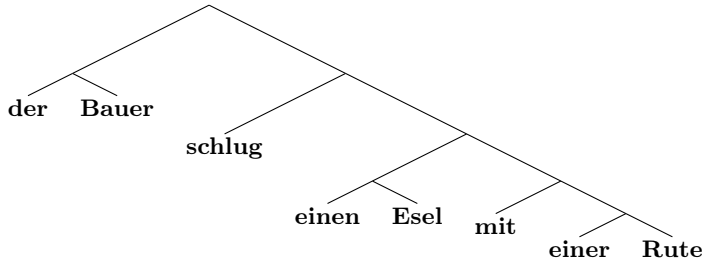
The ambiguity of (8) is due to the fact that the prepositional phrase **mit einer Rute** [≈ with a cane] can either apply to the noun **Esel** [≈ donkey][10] or to the verb phrase **schlug einen Esel** [≈ beat a donkey]. (8) accordingly has two different constituent structures (10a) and (10b), corresponding to the two readings (9a) and (9b):

(10) a.



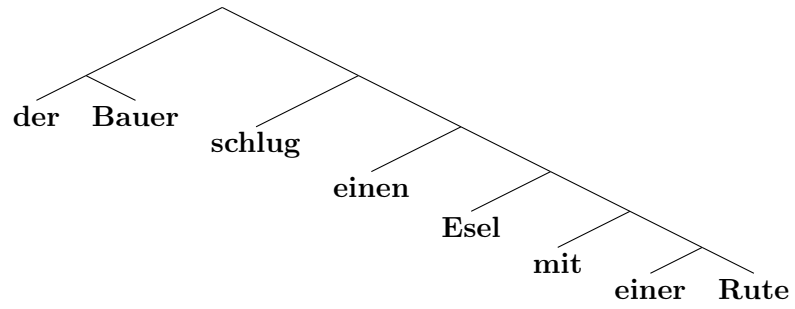
[9]Here is an analogous case involving lexical ambiguity. Suppose Fritz first lives across the street from a bank but there are no benches anywhere in the area. He then moves to an apartment that is across the street from a park bench, though the nearest bank is far away. In this case the sentence **Zum zweiten Mal wohnt Fritz gegenüber einer Bank** [≈ For the second time Fritz lives across the street from a bank/bench] is not true – no matter whether the form **Bank** [≈ bank/bench] is supposed to stand for benches or banks; and one cannot (seriously) have it refer to both.

[10]... or to the determiner phrase **einen Esel** [≈ a donkey]:



However there are semantic reasons that speak against this structure; we will return to this in Chapter 6.

b.



Term evaluation

This type of ambiguity can also be found in mathematical terms or formulas; to cope with it, bracketing has been invented, which makes a direct connection with the corresponding reading of the term. We will now take a closer look at this connection; this will not only help us to get a better understanding of the phenomenon of structural ambiguity but of the interpretation of complex expressions in general. We will use an arithmetic example that does not presuppose any semantic knowledge. Only from Chapter 2 onward will we see how these considerations carry over from the 'language' of arithmetic to natural language. No reason to be afraid of too much mathematics, though: we will not be concerned with calculating numbers but with the question how *designations* of numbers are understood.

First of all, one needs to notationally distinguish between numbers and number designators – *terms*, for short. In analogy to natural language examples we will thus write terms in **boldface** when we talk (i.e., write) about them, while we will refer to numbers with ordinary sequences of digits set 'normally'. Thus, e.g., $\mathbf{9}$ and $\mathbf{3^2}$ are two distinct terms that both stand for the number 9. The number for which a term stands will, from now on, be called the *value* of that term. To refer it, we will put double brackets around the term. Thus the value of $\mathbf{3^2}$ is: $[\![\mathbf{3^2}]\!] = 9 = [\![\mathbf{9}]\!]$, although, clearly, $\mathbf{9} \neq \mathbf{3^2}$; the terms are distinct but they stand for the same number.

In order to see how terms refer to numbers and which rôle the bracketing plays, we will focus on *power terms*, i.e., terms of the form a^b, where a and b may themselves be power terms (!) or single *digits*: $\mathbf{0}$, $\mathbf{1}$ ...,$\mathbf{9}$; sequences of digits, like $\mathbf{99}$, will be ignored.[11] Each of the ten digits stands for a number:

(11) *Evaluation of simple terms*
$[\![\mathbf{0}]\!] = 0$; $[\![\mathbf{1}]\!] = 1$; $[\![\mathbf{2}]\!] = 2$; ...; $[\![\mathbf{7}]\!] = 7$; $[\![\mathbf{8}]\!] = 8$; $[\![\mathbf{9}]\!] = 9$.

It is worth pondering the above equations for a moment. At first blush they may appear circular, since they seem to say that boldface and normal digits refer to the same number. However this impression is misguided! For the equations only make a statement about what the boldface digits refer to.

[11] We will return to them in the exercises.

The other digits are not talked about, they are only *used* to make reference to the corresponding numbers Had we wished to talk about both types of digits, we would have had to write the following instead of the first equation:

(12) $\quad [\![\mathbf{0}]\!] = [\![\text{`0'}]\!]$

In (12) '0' acts as a name for a digit that is used in the equations in (11), where it refers to the number 0. As a statement about what the digit **0** refers to, (12) is indeed circular. For, like boldface types, inverted commas are are but a means to designate the number in question. Thus viewed, (12) says the same as:

(12′) $\quad [\![\mathbf{0}]\!] = [\![\mathbf{0}]\!]$

(12″) $\quad [\![\text{`0'}]\!] = [\![\text{`0'}]\!]$

On the other hand, the equations under (11) make a connection between symbols (the digits) and objects for which these symbols stand (numbers). What is confusing about this is merely that we are talking about the very symbols that we are using. We have to deal with this kind of *apparent* circularity all the time when we are talking about the meaning of German expressions in German (as in the original version of these class notes, but not this very translation though!).

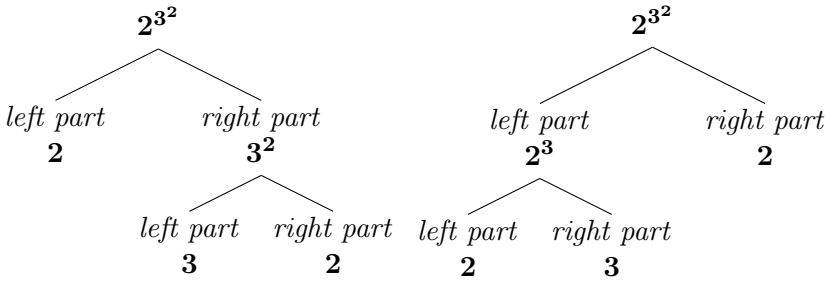
Even if the equations in (11) are not circular, they are still pretty trivial. The fact that **6** stands for the number 6, is known to everyone. How do we know this? Because we have learnt it some time ago – usually in primary school or shortly before that. And not only this: the digit **6** refers to the number 6 by definition, so to speak, it is the common designation of that number, and we do not know this symbol other than as a designation of that number. In this sense the equation '$[\![\mathbf{6}]\!] = 6$' is trivial. On the other hand it does not express a compelling state of affairs. One may, after all, imagine that the symbol **6** designates the number 5;[12] and there is certainly nothing contradictory in this idea. The trivial equations thus are not tautologies, i.e., statements that become contradictory when negated – like the equations in (12′) and (12″).

Let us now turn to the complex terms. In mathematics notations like $\mathbf{2^{3^2}}$ are avoided because they are ambiguous, and brackets are used instead: $\mathbf{2^{3^2}}$ can be disambiguated as $\mathbf{(2^3)^2}$ and then denote the number 64 [$\approx 8^2$]; or else the bracketing is $\mathbf{2^{(3^2)}}$, and we are dealing with the number 512 [$\approx 2^9$]. Without the brackets, one would not know which of the two possibilities is intended. What precisely does the bracketing indicate? This is easy: *the bracketing dissects a term into its parts*, which themselves are terms and

[12] ... just like the oldest decimal system, the North Indian Brahmi notation (3rd century BC) uses practically the same symbol for the number 8 like the West Arabic Gobar system (11th century AD) uses for 5 (viz. something like Ч).

which themselves can be dissected into their parts (as long as they are not single digits). When bracketed as $\mathbf{2^{(3^2)}}$, the term consists of the parts **2** and $\mathbf{3^2}$ (*in this order*); with the bracketing $\mathbf{(2^3)^2}$, on the other hand, it consists of $\mathbf{2^3}$ and **2**:

(13) a. b.



If one wants to determine the number for which the whole term stands, i.e., its value, one follows the bracketing in that one first determines the values of its parts. Thus in (13a) one first needs to find out what the terms **2** and $\mathbf{3^2}$ stand for, and for (13b) one needs the values of $\mathbf{2^3}$ and **2**. Once this is known on puts the value of the left term part (the base) to the power of the value of the right one (the exponent):

(14) a. $[\![\mathbf{2^{(3^2)}}]\!] = [\![\mathbf{2}]\!]^{[\![\mathbf{3^2}]\!]}$ b. $[\![\mathbf{(2^3)^2}]\!] = [\![\mathbf{2^3}]\!]^{[\![\mathbf{2}]\!]}$

(14a) and (14b) look complicated at a first glance and circular at a second. Both impressions are deceptive. If one finds (14a) opaque, one should try a verbalized version: *The value of the term x ensues if the value of the left term – i.e., the value of* **2** *(i.e.,* $[\![\mathbf{2}]\!]$*) – is put to the power of the value of the right term – i.e., the value of* $\mathbf{3^2}$ *(i.e.,* $[\![\mathbf{3^2}]\!]$*), thus putting* $[\![\mathbf{2}]\!]$ *to the power of* $[\![\mathbf{3^2}]\!]$. As an exercise one may immediately verbalize (14b) too! This ought to help understand the two equations. But it is precisely then that one gets the impression that they are completely void of content. To see that this is not so, we may look at the general pattern underlying (14a) and (14b):[13]

(15) *Evaluation of complex terms*
If a and b are terms, the following holds: $[\![a^b]\!] = [\![a]\!]^{[\![b]\!]}$

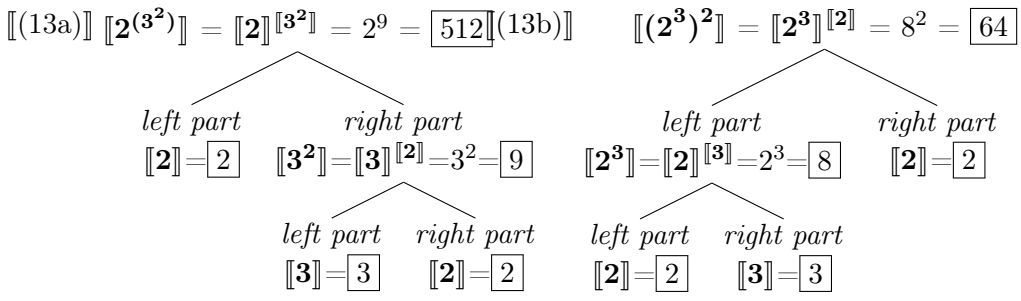
The fact that superscripted terms are to be understood as in (15), is not compelling in that it could have been otherwise. For instance, a^b may also have denoted the product of the number denoted by a with the value of b. In that case one would have needed the following equation instead of (15):

(15′) $[\![a^b]\!] = [\![a]\!] \times [\![b]\!]$

[13] *Question*: Why are we using italics for terms in (15) and not boldface as elsewhere? – *Answer*: Boldface is used to denote *single* terms whereas 'a' and 'b' are *variables* for *arbitrary* terms. If we had used '**a**' and '**b**' instead, we would have said something about the letters 'a' and 'b'.

It should be noted that the symbol '×' in (15′) stands for an arithmetic operation, viz. multiplication. Similarly, in (15) superscripting to the *right* of the equality sign stands for the exponentiation of two *numbers*, viz. the values of the terms a and b. Superscripting to the *left* of the equality sign, on the other hand, stands for a configuration of symbols (the terms). No arithmetical operation would make sense here, for they apply to numbers, which can be multiplied, exponentiated, etc. Terms can only be written next to each other, on top of each other, etc. The equation in (15) thus says that a particular combination of number symbols – to wit, writing them next to one another while superscripting the right one – corresponds to a certain arithmetical operation – viz., exponentiation. And there is nothing circular in this correspondence; rather, it is a common, but neither necessary nor universal notational custom. Due to the prevalence of that custom, (15) appears almost as natural as (11). That superscripting stands for exponentiation – which is precisely what (15) says – is something we learnt at school after all, though not quite as early as the interpretation of the digits. And the seeming circularity of (15) is of the same ilk as the one perceived in (11). Just as we referred to the numbers with the common ('Arabic') digits in (11) that the equations were about, so in (15) the exponent is represented by superscripting – which is the notation that the equation is about. The use of superscripts to the right of the equality symbol is, of course, presupposed in (15). Hence the equation can only be grasped by someone who already knows this and thus knows what exponentiation is, i.e., which arithmetic operation it is. So (15) is not a rule of *computation*, but merely an explanation of the exponent *notation*, which such computational rules always presuppose.[14]

The equations in (11) and the pattern in (15) suffice to evaluate bracketed terms like (13a) and (13b). For according to (15), the values of the complex terms are derived from the values of their parts, which themselves are evaluated according to (15) (if they are complex) or (if they are digits) have their values declared in (11). This *term evaluation* of (13a) and (13b) can again be represented by way of a tree:



[14]The following rule is a case in point: $n^m = \underbrace{n \cdot \ldots \cdot n}_{m\text{-times}}$ – or more precisely: $n^0 = 1$; $n^{m+1} = n \cdot (n^m)$

The evaluations of (13a) and (13b) stepwise follow the bracketing – from the smallest parts to ever larger ones. The values of the smallest parts are given in advance in the equations (11); for compound terms the general rule (15) comes into play. Obviously, arbitrarily complex terms can be evaluated in this way – as long as their smallest parts are single digits and they are only combined by superscripting.

Compositionality

The reason why we discussed the procedure of evaluating terms as illustrated in (13a) and (13b) so extensively is that it plays such a central rôle in logical semantics. In order to account for the meanings of complex linguistic expression, we will always assume that they come with brackets, so to speak, i.e., that the are decomposable into parts and parts of parts. As a rule the precise bracket structure is motivated syntactically, mostly coinciding with a simplified surface structure – as in the case of (10a) and (10b) above. At times, however, we need to apply a different, purely semantic bracketing, a so-called *Logical Form*. We will make this explicit when we get to analyzing pertinent examples. For the time being, we only note that the interpretation of complex linguistic expressions always requires a decomposition into parts, whatever the source of this decomposition may be.

The analogy between term evaluation and interpretation of natural language expressions has its limits. After all, expressions normally do not stand for numbers that one could apply arithmetical operations to. But the overall procedure is the same. In semantic theory, this is captured by a methodological principle:[15]

(16) *Principle of Compositionality*
The meaning of a complex expression derives from the meanings of its parts and the relevant syntactic construction.

This is supposed to mean that the interpretation of a sentence follows constituent structure – just like the term evaluation in (13a) and (13b). So in order to determine the meaning of (8), one first has to decide on one of its two readings. In the case of (10a), then, the meaning of the predicate ensues by combining the meaning of **schlug einen Esel** [≈ beat a donkey] with that of **mit einer Rute** [≈ with a cane].

[15]The origins of this principle are unclear. It used to be called *Frege's Principle*, after the founder of logical semantics, but: "Frege selbst hat das Prinzip niemals ausdrücklich aufgestellt, es wurde nur nach ihm benannt, weil es gut zu seinen sprachphilosophischen Grundgedanken passt" [≈ Frege himself has never explicitly formulated the principle, it was only named after him because it fits his basic ideas in philosophy of language so well] (as Irene Heim put it in her 1977 Master's Thesis). In its modern form as a generalization of term evaluation the principle of compositionality (though not under this name) appears in the essay *Universal Grammar* (1970) by Richard Montague.

Unlike term evaluation, the Principle of Compositionality is not concerned with numbers, but with linguistic meanings. We will see what these meanings are as we go along. We already mentioned that meanings cannot be combined by applying arithmetical operations – so exponentiation won't do it. But we will introduce *semantic operations*, which combine meanings instead.

Apart from the parts of complex expressions and their meanings, the Principle of Compositionality also makes reference to syntactic constructions. These are of importance in that they determine just *how* the meanings of the parts are to be combined. So far this is hard to see – we have not yet come across any semantic operations after all. The following example might still offer a glimpse:

(17) **Fritz und Eike sind verheiratet.**
[≈ Fritz and Eike are married.]

This sentence is ambiguous (cf. the exercise). Its two readings may be paraphrased as follows:

(17′) **Sowohl Fritz als auch Eike sind verheiratet.**
[≈ Both Fritz and Eike are married.]

(17″) **Fritz und Eike sind miteinander verheiratet.**
[≈ Fritz and Eike are married to one another.]

It is not entirely obvious what the source of the ambiguity should be. Does the word **und** [≈ and] have two meanings? Or does the sentence (17) have two syntactic structures? One reason that speaks against homonymy is that this would have to occur a lot of languages: Englisch **and**, French **et**, Hebrew **ve**, ... all trigger the ambiguity observed in (17). This can hardly be accidental. But then homonymies are usually products of coincidence; and that the same homonymy occurs in two unrelated languages is extremely improbable. Furthermore, the ambiguity in (17) can also be found with other plural determiner phrases (like **die beiden** [≈ the two]) where it cannot be explained in terms of an ambiguity of the word **und** [≈ and]. This in itself does not speak in favor of structural ambiguity,[16] but it lets it appear more plausible. However, if (17) is structurally ambiguous, this is arguably due to the way in which the subject **Fritz und Eike** [Fritz and Eike] is combined with the predicate **sind verheiratet** [≈ are married]: according to one construction, (17) is understood in the sense of (17′), according to the other one it comes out in the sense of (17″). The Principle of Compositionality then says that in both cases the 'input', the meanings of subject and predicate, is the same, but that the two constructions combine this input in different ways.

[16] Systematic polysemy is another option.

The Principle of Compositionality refers to the *parts* of complex expressions. As in term evaluation, these are always supposed to be the *immediate* parts – not the parts of parts. This is an important restriction. For without it the principle would only say that the meanings of complex expressions can be obtained from those of their parts and parts of parts as well as their syntactic structure. Since among the parts of parts of a complex expressions are the words that occur it it, this would boil down to saying that the word meanings together with the syntactic structure determine the meaning of the overall expression. This is certainly right, but far too little. Rather, compositionality means that the meaning of each and every expression emerges directly from the meanings of its immediate parts (and the syntactic construction they stand in). In this way the meaning of the whole can be determined step by step, starting with the word meanings and all the way up to the complete sentence. At this point the difference may appear minimal, but we will later see that it must not be neglected.

As is well known, linguistic expressions are unlimited in their number and complexity; and in principle, speakers know their meanings even if they have never heard or read them before. The Principle of Compositionality explains how this is at all possible: in order to understand a complex expression, one only needs to know what the words it contains mean and which semantic operations correspond to the syntactic constructions it employs. Though the number of words and of syntactic constructions is very large, it is not unlimited. It is thus principally possible that one learnt each of them individually and brings them into action in a compositional way. A glance at the evaluation of terms again shows how one may conceive of the semantic knowledge underlying linguistic understanding as coming in two parts. One part consists of *lexical rules*, that say what the meaning of each word is – like the equations in (11) define the value of each single digit; the other part consists of *grammatical rules*, that say how each construction combines the meanings of the expressions that undergo it – like the scheme (15) says that the value of two terms written next to each other (where the second one is superscripted) results by exponentiation of the values of these two terms. Since this is a course on logical semantics, lexical rules will be largely neglected. One cannot do totally without them though – compositionality has got to start on something after all; but they will be mostly trivial, though not quite as trivial as the equations in (11).

Since according to the Principle of Compositionality, the parts of expressions only contribute their own meaning, they may always be *substituted* by synonymous expressions without thereby affecting the meaning of the whole. In semantics (and philosophy of language) this immediate consequence is known as the:

(18) *Substitution Principle*[17]
If one part of an expression is replaced by a synonymous expression, the meaning of the whole expression does not change.

As we will see more often during this course, typical applications of the Substitution Principle are negative: if two supposedly synonymous expressions cannot always be substituted for each other without changing the meaning of the whole, they could not have been synonymous in the first place. Let us illustrate this with an example. In view of sentences like (19a) and (19b) one may at first suspect that the prepositional phrases **in einer Woche** [≈ in a week] and **heute in einer Woche** [≈ in a week from today] have the same meaning:

(19) a. **Kommen Sie in einer Woche noch mal vorbei.**
[≈ Come back again in a week.]
b. **Kommen Sie heute in einer Woche noch mal vorbei.**
[≈ Come back again in a week from today.]

(19b) may be slightly more precise as an order, but obviously boils down to (19a). In this case it appears that one expression may be replaced by the other without changing the overall meaning.[18] That the two sentences are not synonymous after all, can be seen from the following substitution:

(20) a. **Der Techniker bat mich, in einer Woche noch mal vorbeizukommen.**
[≈ The technician requested that I come back again in a week]
b. **Der Techniker bat mich, heute in einer Woche noch mal vorbeizukommen.**
[≈ The technician requested that I come back again in a week from today]

If said technician had told me (19a) or (19b) yesterday, (20a) would be a correct account of his request – but (20b) would not. Thus (20a) and (20b) cannot possibly have the same meaning.[19] Bur since (20b) results from (20a) by substitution of the singly underlined part by the doubly underlined one, given the Substitution Principle they cannot be synonymous – contrary to

[17] The Substitution Principle already plays a central rôle in Gottlob Frege's essay *Über Sinn und Bedeutung* (1892).

[18] It may be noted that this comparison does not even make use of the Substitution Principle, which makes no claim about substitutions in which the meaning of the entire expression does not change. This may come as a surprise, but becomes clear by the following reformulation of the principle (or *contraposition*, as logicians call it): *If substitution of a part of an expression by another one changes the meaning of the whole expression, the two parts are not synonymous.*

[19] This argument also makes use of a very general principle that we will only encounter in the next chapter: the *Most Certain Principle*.

the first impression based on (19a) and (19b). So much for the illustration of the Substitution Principle to which we will return shortly.

The Principle of Compositionality has proved extraordinarily helpful in logical semantics – among other things because it helps finding out what the meanings of linguistic expressions are. (More about this from Chapter 2 onward.) We will therefore adhere to it throughout this course and only regard a semantic analysis as complete if it accounts for the meanings of the expressions under scrutiny in a compositional way.

0.4 Exercises for Chapter 0

A1 In what way do ambiguous expressions present a problem for the application of the criterion of Invariance? And how can the problem be solved?

A2 Though the verbs **telefonieren** [≈ make a phone call] and **anrufen** [≈ call over the phone] describe similar activities, they do no have the same meaning.

(a) What is the difference in meaning?

(b) Is one a hyponym of the other?

(c) In what way do the meanings of the verbs make a different contribution when it comes to determining an unnamed partner of a telephone conversation?

[Only look at the *intransitive* uses of these verbs!]

A3 In (5c) the converse relation between **älter** [≈ older] and **jünger** [≈ younger] survived an expansion by **zwei Jahre** [≈ two years]. Show that not every modification of converses again leads to converses. Hint: Consider nouns instead of comparatives!

A4 Show that the two uses of **Glas** [≈ glass] – material vs. container – do constitute two readings, wich behave like two distinct words.

A5 Formulate a compositional term evaluation for arabic number terms consisting of more than one digit. Assume a 'left-branching' bracketing for the purpose – e.g., **(((42)5)1)** for **4251**.
Extra question: What would be the problem with a right-branching structuring like **4(2(51))**?

A6 Formulate rules of term evaluation for multiplication and addition that correctly accounts for terms like $\mathbf{(5 \cdot (3 + 2))}$ and $\mathbf{((0 \cdot 6) + (2 + 3))}$.

A7 Show that the sentence **Fritz und Eike sind verheiratet** [≈ Fritz and Eike are married] is ambiguous.

Chapter 1

The Meanings of Sentences

According to the Principle of Compositionality the meanings of complex expressions are determined by the meanings of their parts, which are in turn determined by the meanings of their parts, etc. – down to the smallest parts, which are usually words. In order to account for the meaning of a complex expression, one can start with the meanings of the words it contains, climbing up compositionally to ever more complex parts until one finally reaches the meanings of the immediate parts of the whole expression and combines them to determine its meaning. We will later encounter this procedure in many examples. However, before we get there, we need to find out just what meanings are. In order to do so, the reverse strategy – from complex expressions to words – is recommendable. More precisely, we will start by identifying the meanings of complete *sentences*. This will be done by way of examples, because in the beginning we only want to know what *kind* of objets sentence meanings are, and not so much how the meaning of one examples differs from that of the next. Once we have a better grasp on the meaning of sentences in general, we will, in the next chapter, address the question how it is determined from the meanings of its parts and what those meanings themselves are.

1.1 The Principal Principle of Semantics

One difference between someone who hears (or reads) a sentence like (1) and understands it, and someone who does not understand it, is that it can create a certain image, an inner picture, for the former:

(1) **May maliit na batang babae na naglalaro.**

(1) is a sentence of Tagalog,[1] which more or less literally translates as:

[1]Tagalog (stress on the 2nd syllable), the official language of the Philippines since 1962, is an Austronesian language that is spoken by some 40 million native speakers across

(1′) **Ein kleines Mädchen spielt.**
[≈ A little girl is playing.]

As a reader of these notes you understand (1′) (or at least the gloss), whereas you are probably at a loss with (1). And when you first read (1′), a certain image may have entered your head. This may suggest the following answer to the question what sentences mean: *The meaning of a sentence is the idea it creates.* There are many reasons that speak against such a notion of meaning, though. For the ideas associated with sentences are *(a)* too *subjective* to serve as meanings, they are *(b) restricted* to few kinds of sentences, they are *(c) irrelevant* for communication, and on top of that they are of a *(d) private* nature:

(a) Different speakers associate different things with the same expressions, which they still use in the same sense, with the same meaning.

(b) In the case of (1′), associated internal pictures may be conceivable meanings, but how, e.g., could the meaning difference between **Pizza esse ich zweimal im Monat** [≈ I eat pizza twice a month] and **Pizza esse ich jede Woche** [≈ I eat pizza every week] be captured in inner images?

(c) Due to personal experience, speakers may make all kinds of associations a sentence which do not affect the way they understand it or its meaning.

(d) The ideas and associations of an individual are principally inaccessible to other speakers, so how could they serve to communicate between speakers?

Given these objections, it is advisable to look for a less naive notion of meaning. The following observation turns out to be useful to this end. Let us compare (1) with another sentence of Tagalog, the translation of which will only be revealed later:

(2) **May maliit na batang lalake na nagaawit ng awit.**

As I am writing this, a little girl who is busy with a jigsaw, is sitting next to me in the train. So sentence (1) was not chosen at random, but was actually true at the time of writing. The same holds of (2). I put it down shortly afterwards (when the girl was still playing), and it was also true to the facts. But in the few seconds that have passed since then, this has changed: unlike (1), (2) is no longer true.

the world – one of them being my former colleague Tim Fernando (now Trinity College, Dublin), to whom I owe the examples.

You do not know what (2) means, but one thing you do know now is this: (1) and (2) do not mean the same. They cannot, for (1) is true in the given circumstances, while (2) is not. We note this elementary conclusion as the:

(3) *Most Certain Principle*
If in the same circumstances, one of two sentences is true while the other is not, then they do not have the same meaning.

We already employed this principle at the end of the preceding chapter. But its relevance lies not so much in its actual application but in the fact that it is an undisputed, even banal observation on a subject about which we have almost no other secured pre-theoretic knowledge.[2] However blurry and opaque the phenomenon of meaning may appear, the connection made out in the Most Certain Principle is quite obvious – and this connection holds between the meaning of sentences and their truth. So whatever meanings (at least of sentences) may be, they have something to do with truth.

The above example illustrates that truth is not absolute in that the same sentence can be true or false (= not true), depending on what it relates to: sentence (2) correctly describes the circumstances in which I had initially been; in relation to the subsequent scene, however, it is false. In semantics, the circumstances to which a sentence may relate are called *situations of* which it is true (or false), *to* which it *applies* (or does not apply, as the case may be). We had looked at two situations that differed with respect to time[3]; and (1) and (2) were both true of the first, earlier situation, while (2) was false of the second situation. It is worth noting that this situational reference is *implicit* in that there is no part of the sentence that denotes the situation.

According to the Most Certain Principle, two synonymous sentences must have the same *truth conditions*, i.e., they must both be true or both be false of any given situation. Following the tradition of logical semantics (aka as *truth-conditional semantics* indeed), we will make no further assumptions about sentence meaning and for the following (until further notice) only investigate those meaning aspects of sentences that are reflected in their truth conditions. We use *content* as a cover term for these aspects. The content of a sentence thus comprises those parts of meaning that make up its truth conditions and may thus be characterized as follows:[4]

(4) *Principal Principle of Semantics*

[2]The term (*Most Certain Principle*) is due to logician, semanticist, and philosopher of language Maxwell J. Cresswell from New Zealand, who formulated the principle in his essay *The Autonomy of Semantics* (1982).

[3]Even though they were set in the same train compartment, they also differed with respect to space, since in the meantime the train had moved!

[4]The Principal Principle of Semantics ultimately goes back to the *Tractatus logico-philosophicus* (1921) by the Austrian philosopher Ludwig Wittgenstein.

> That two sentences have the same content means that they apply to exactly the same situations.

This formulation does not mention the situations of which the sentences in question are *false*. In fact, this is not necessary. For if two sentences apply to exactly the same situations, i.e., if they are true of the same situations, then they are both false of all other situations.

The Principal Principle of Semantics merely regulates how the term *content* is to be used when applied to sentences. It only becomes effective by restricting the analysis of sentence meaning to content. This restriction on the description of sentence meaning has proved fruitful for semantic analysis. Only in Chapters 8–10 [yet to be written] will we see where this methodology meets its limits. Until then we will content ourselves with analyzing sentences so that it suffices for them to come out equivalent if they are both true or false in arbitrary circumstances.

It follows from the Principal Principle of Semantics that two sentences that differ in content cannot be true of exactly the same situations. Any semantic difference between two sentences must therefore be reflected by (at least) one situation to which only one of them applies. In the case of (1) and (2), which indeed differ in meaning, we had taken the existence of such a situation as indicating a difference in meaning[5] The Principal Principle of Semantics now forces us to come up with such evidence for *any* semantic difference between two sentences. We will see that finding such evidence is not always easy. But then the Principal Principle of Semantics turns out be so useful that the search will always be worth the effort.

1.2 Propositions

According to the Principal Principle of Semantics the content of a sentence only concerns which situations it applies to. One may thus identify the content of a sentence with these situations. This is exactly what we will do. The content of (5) thus consists of precisely the situations in which someone is coughing:

(5) **Jemand hustet.**
[≈ Someone is coughing.]

If, e.g., Tom is sitting at his desk and suddenly needs to cough, then (5) applies to this situation; if, however, the entire audience during a concert is quiet, then this would be a situation to which (5) does not apply. Let us call the first situation t and the second one k. Then according to our assumption the content of (5) comprises the situation t, but not the situation k. It should be noted that this is so whether or not (5) itself is uttered in the situations

[5] By the way, (2) says that a little boy is singing.

at hand. The sentence need not be uttered at all and applies to t but not to k. This is just a matter of its meaning, its content.

That the content of a sentence *consists* of situations is supposed to mean that these situations *as a whole* form the content of the sentence. We will thus conceive of the content of a sentence as a *set*, in the mathematical sense of the term. The defining feature of a set is that it collects arbitrarily many objects – in our case: situations – into one abstract whole. The objects thus put together are called the *elements* of the set. Elementhood is thus a relative concept: a thing, or a situation, is never an element *per se*, but can only be an element *of* a given set. As a case in point, the above situation t is an element of the content of (5), but not an element of the content of:

(6) **Niemand hustet.**
[≈ No one is coughing.]

What is a mathematical concept like set doing in logical semantics in the first place? Wouldn't it suffice to use a 'naive' concept of wholes when defining the contents of sentences? Yes and no. As long as we are exclusively dealing with *sentence* contents, mathematical modeling is rather pointless. But we are going to find out how the meanings (i.e., the contents) of sentences compositionally emerge from those of their parts. And for this, some elementary logical and mathematical tools turn out to be indispensable. This will already become clear in the next chapter. Apart from this, the mathematical account of the concept of meaning also introduces a certain amount of precision that prevents difficult problems being talked away by flowery style. Instead it makes it easier to formulate theoretical hypotheses that are in principle testable. But before we get to this, we need to understand some of the necessary mathematical tools, starting with the concept of a set. For even though we will not present an introduction to set theory here – this won't be necessary for our purposes anyway – a minimum of background knowledge is mandatory. For the beginning, we will content ourselves with a crucial property of sets that we will again formulate in terms of a general principle. It says that a set is defined solely by the objects that are collected in it:

(7) *Principle of Extensionality*
If a set A has exactly the same elements as a set B, then $A = B$.

The gist of the Principle of Extensionality is best seen by way of an example. If one collects all German towns with over a million inhabitants into one abstract whole, one obtains a set with four elements. We can write this set as '{Berlin, Hamburg, Munich, Cologne}', listing its elements in curly brackets.[6] In this case we ordered the elements by their number of inhabi-

[6]Of course, it is not the elements themselves that are listed, but *denominations* of these elements, in this case *names* of cities. Incidentally it is not the *kind* of denomination that

tants. We could just as well have presented them in a different order, e.g. as '{Hamburg, Cologne, Berlin, Munich}'. *According to the Principle of Extensionality* this must be the same set. For the same reason, some (or all) elements could be listed more than once without any effect on the set itself. So we have: {Berlin, Hamburg, Munich, Cologne} = {Hamburg, Cologne, Berlin, Munich, Berlin, Hamburg, Cologne}. In particular, from the length of the list in curly brackets one cannot deduce the number of elements of the it represents.

The Principle of Extensionality can neither be proved nor refuted. Rather, it is a conceptual assumption: the concept of a set, i.e., the concept of a collection of objects into one abstract whole, is to be understood so as satisfy this principle. Should it turn out that two objects A and B have the same elements without being the same – i.e. $A \neq B$ – then A and B could not be sets in the set-theoretic sense (and one should thus not have talked of *elements* in the first place).

The Principle of Extensionality guarantees that the Principal Principle of Semantics can be cashed in by regarding sentence contents as sets of situations: according to the Principle of Extensionality, the sets of situations of which two sentences are true, coincide just in case the two sentences apply to exactly the same situations, i.e., if they have the same content according to the Principal Principle. We will come across further set-theoretic principles during the next chapter.

The contents of sentences are sets whose elements are exclusively situations. In semantics such sets are called *propositions,* and it is said that a sentence *expresses* the proposition that is its content.[7] We follow this terminology here and record this in the form of terminological regulations (*definitions*):

D1.1 A *proposition* is a set of situations – i.e., a set all of whose elements are situations.

D1.2 That a sentence S *expresses* a proposition p, means that p is the content of S.

Warning: not every proposition needs to be the content of a sentence! As one can imagine, contents of sentences are usually rather large sets of situations. Thus, e.g., the above situation t is only one of the many situations of which (5) is true. On the other hand, sets of situations can be arbitrarily small and thus, e.g., only have one or two elements. As the exercises will bring

counts as long as the same object is denominated. The above set could also have been written as '{Berlin, Hanseatic City of Hamburg, Bavaria's capital, Willy Millowitsch's birthplace}'.

[7] It should be noted that, according to D1.2 *sentences*, i.e., linguistic expressions, express something; in a different manner of speaking, mainly used and to be defined in pragmatics, it is *speakers* that express something by using expressions.

out though, there does not seem to be any sentence whose content is such a small set of situations. To put it shortly: the concept of a proposition is more general than that of sentence content in that sentence contents are always propositions but not every proposition is the content of a sentence.

1.3 Situations

We have seen that a sentence can relate to a concretely given situation. By uttering the sentence the situation is characterized as an element of the proposition the sentence expresses. This characterization may serve different purposes – to answer a question, to clear up a misunderstanding, or merely to inform the audience of a state of affairs. How the proposition expressed by a sentence is brought into use in communication is a question of pragmatics that we are not concerned with here. It is also a question of language use which situation the speaker refers to when he says something about it. Usually it is a situation in which the speaker himself is currently in. If at the beginning of a concert Hans expresses his surprise by whispering (6) to the woman sitting next to him, he may thereby refer to the first few minutes of the concert. And he may be right, even though at the same time a cloakroom attendant is coughing in the foyer. The foyer then simply does not belong to the situation Hans is talking about. Though there is a larger situation that also includes the foyer, that would not be the situation Hans is referring to. To be sure, he is in both situations but he only refers to one of them, the smaller one.

We thus assume that situations come in different sizes, both spatially and temporally. It is therefore natural to identify situations with spatiotemporal regions. We will do so indeed – though only until the end of Chapter 9 [which is yet to be written]. More exactly, we define:

D1.3 A *situation* is an *arbitrary* connected spatiotemporal region.

Though not much hinges on this precisification (or *reconstruction*) of the common notion of a situation, it helps making the building blocks of sentence meanings more tangible; at times we will also call them in to decide problematic cases.

As spatiotemporal complexes, situations may not just differ in size but also include or overlap each other. Thus the above-mentioned extended concert-cum-foyer situation includes the situation to which Hans refers in our example. Since he is in both situations, i would not make much sense to say that Hans is talking about *the* situation that he and his seat neighbor are in. In fact, they both are in a lot situations together and at the same time: in the concert hall during the first five minutes; in Row 17 during all of the concert; in the Amsterdam *Concertgebouw* during the first movement of the Alpine Symphony; in early 21st century Europe; etc. All these situations

overlap, and in some of them (5) is true, while (6) is true in others. But only one of them is what Hans refers to with his whispered utterance of (6). Which situation this is depends on his intentions. If he wants to be understood by his neighbor, it better be obvious to her what he refers to. Thus if Hans had earlier expressed his fear that too many coughs may spoil the concert for him and now five minutes have passed since its beginning, it is indeed natural to interpret Hans's utterance of (6) as relating to the situation in the concert hall during these first five minutes. Under different circumstances though – perhaps if his neighbor had just enquired about the health state of his family – he may have referred to a different situation.

It is again a general question of pragmatics how speakers manage to make situational reference comprehensible to their audience. In many cases, though, it is not really important *precisely* which situation the speaker means to talk about, as long as it is within reasonable bounds. Whether Hans refers to the first five minutes or the first four and a half minutes, to the entire music hall or the front stalls only – these details are obviously irrelevant for communication and are not even noticed by speaker or hearer.[8] Moreover, the content of the sentence uttered, the proposition it expresses, may largely *neutralize* situational reference. Sentence (8) is a case in point: for – contrary to the first impression it may create – it does not matter to which place it is meant to relate:

(8) **In den USA gibt es in jedem Bundesstaat eine Stadt namens *Springfield*.**
[≈ There is a place called *Springfield* in every US state.]

Unlike (6), where implicit reference is made to the spatial location of the situation, (8) explicitly mentions a particular place, viz. the USA. Let us suppose that Hans utters (8) while waiting for the tram after the concert. One may perhaps think that he is thereby talking about a distant, spatially quite extensive situation. But this need not be so. After all, what is said by (8) is that, *at some, not explicitly mentioned, time*, each US-state contains a Springfield. As in (5) and (6), the time to which Hans's utterance relates is again that of the situation about which Hans is talking – the ten minutes at the Amsterdam tram stop maybe. In (8) no further reference to this situation is made. In particular, its spatial location is not referred to. There is a place that gets mentioned, but this is independent of the situation (the utterance of) the sentence is about. Reference to the place of the situation has been *neutralized* by the use of the explicit local specification **in den USA** [≈ in the USA]: (8) is equally true (or equally false) of any situation that is temporally located within the 10 minutes mentioned as it is true (or false)

[8] This vagueness of what is meant is again to be accounted for in pragmatic analysis, which is not at all easy.

of the utterance situation.[9] Hans may have uttered (8) after the concert to say something about a situation in which he and the person he is talking to happen to be. What he would have said then, would have been that this situation is located at a time at wich any US-state contains a place called *Springfield*. Since the situation expressed by (8) consists of all situations that satisfy the underlined condition, the situation Hans is talking about would even be in this set if it only contained the first two stall rows in the concert building – as long as there are only enough Springfields in the USA (which there aren't, by the way). Quite generally, the sentence (8) is true of every situation that temporally coincides with some situation to which it applies. In this sense the spatial aspect of the situation a speaker uttering (8) refers to is neutralized – which in turn means that the hearer need not know down to the smallest (spatial) detail to which situation the speaker is referring to.

What is true of space, also applies to time. It too may be largely neutralized:

(9) **Am 22. April 2002 haben insgesamt fünfzehn Leute angerufen.**
[≈ On April 22, 2002, altogether fifteen people called.]

The sentence can be true of quite different situations: my office, my elder son's room, the complaint department of a computer dealer, etc. But if it is true of such a situation, time practically plays no rôle.[10] Let us consider my younger son Tom's room on April 23, 2002. If (9) applied to this situation t (this spatiotemporal environment, that is), this means that a day before fifteen people had called Tom; but then (9) is also true of the situation t' that starts a day later and ends two weeks thereafter; for it would also hold of this situation that fifteen people had called there – at Tom's place – *on April 22, 2002*.

We end the section with a general characterization of the concepts of neutrality introduced here, even though they will hardly come back to it in what follows:[11]

D1.4 A proposition p is spatially [or *temporally*] *neutral* iff for all sit-

[9] For factual reasons the temporal restriction need not be taken too narrow; for place names and country borders do not change that often. It thus does not matter whether the situation to which the utterance of (8) is supposed to relate, takes a few minutes (or days) longer.

[10] ... with a minor qualification that will be addressed in an exercise. – Incidentally, (9) is (structurally) ambiguous. In its co-called *individual-related* reading the sentence says that fifteen *different* persons called during the day in question (and no matter how many times each of them called), whereas the *event-related* reading, which we are going to ignore here, only requires there to have been fifteen phone calls, at least some of which may have been made by the same person.

[11] From now on we will frequently make use of the common abbreviation 'iff' for 'if, and only if,' – which means that if the one is true, then so is the other, and vice-versa.

uations s and s' the following holds: if s and s' only differ with respect to their spatial [or temporal] position, then either $s \in p$ and $s' \in p$ – or else: $s \notin p$ and $s' \notin p$.

1.4 Logical Space

According to the Principal Principle, any semantic difference between two sentences must be demonstrable by way of some situation (or more), which does not have to be contemporary though:

(10) **Ein Archaeopterix hebt ab.**
[≈ An archaeopterix is taking off.]

(11) **Ein Flugsaurier setzt zur Landung an.**
[≈ A pterosaurian is preparing for landing.]

Of course, (10) and (11) do not mean the same, even though both of them are false of any present situation. One may still assume that there are *past* situations of which only one of the sentences is true. The fact that such situations are in the past and without any human observer has no effect on the very *existence* of such situations. The situations mentioned in the Principal Principle thus have to be taken in a temporally highly inclusive sense. They also include any future situations that may, e.g., serve to differentiate between (12) and (13):

(12) **Ein Astronaut landet auf dem Mars.**
[≈ An astronaut is landing on Mars.]

(13) **Ein Astronaut landet auf der Venus.**
[≈ An astronaut is landing on Venus.]

As soon as, in the far future, an astronaut is flying to Mars, this voyage – which is a spatiotemporal segment after all (and thus a situation) – will make sentence (12) true, but not (13). Hence – assuming that such a voyage will take place – there are indeed situations with which (12) and (13) can be distinguished in the sense of the Principal Principle of Semantics.

But even if one took all situations, large and small, close by and far away, past, present, and future, they would not suffice to instantiate all differences in content. In fact, (12) and (13) could not be distinguished if no one ever flew further than the moon. For in that case both sentence would be false of *any* situation – and hence, in particular, true of *the very same* situations. According to the Principal Principle of Semantics, they would thus have to coincide in content, which is of course absurd.

We need not speculate about the future of space travel to find examples of this kind. The following two sentence also obviously do not mean the same, though they are true of the same situations:

(14) **Der Entdecker der X-Strahlen starb in München.**
[≈ The discoverer of X-radiation died in Munich.]

(15) **Der erste Physik-Nobelpreisträger starb in München.**
[≈ The first Nobel laureate in physics died in Munich]

X-radiation (*vulgo*: X-rays) is a form of electromagnetic radiation first discovered by Wilhelm Conrad Röntgen, the first Nobel laureate in physics, who died in Munich in 1923. Given this, there is no situation to which (14) applies but (15) does not – so that by the Principal Principle one would have to conclude that (14) and (15) have the same content. The principle thus seems to prove untenable.

However, we will not give up that easily. For even if there is actually no situation of which, say, (14) is true but (15) is not, such a situation *could* well have existed. If, e.g., it had not been Röntgen who received the first Nobel Prize in physics but Thomas Alva Edison, who died in West Orange (New Jersey), (14) would be true here and now, but (15) would be false – and we would have a proof at hand differentiating between the contents of (14) and (15), as demanded by the Principal Principle. That we do not seem to, is apparently due to the fact that – at least so far – by "situations" we have always meant *actual* situations. However, if we extend the notion of a situation to such *possible* situations like the one just described, examples like (14) and (15) could not touch the Principal Principle of Semantics. And it is precisely this extension of the notion of a situation that we herewith perform. So if, here and in what follows, we mention situations, we are not just referring to scenarios that are actually taking place, or took place at some time, or will take place at some time, but to all possible situations whatsoever.

The totality of all possible situations, to which the actual situations belong too, is called *Logical Space*.[12] We assume that it is a – very large – set. Since all elements of this set are situations, Logical Space is a proposition; an exercise will show that it is also a sentence content. But let us first get clear about its elements, the possible situations themselves, and its structure, the relations between its elements!

Like actual situations, possible situations are spatiotemporal regions that may also be arbitrarily extended, both in time and in space. Yet unlike actual situations not all possible situations are segments of reality. Rather – if they are not real – they are segments of other realities or, as we will say (following a philosophical tradition): segments of other *possible worlds*. Just as our world from Big Bang until the end of the universe is a gigantic, all-embracing situation, so is the above scenario with Nobel laureate Edison part of another gigantic, all-embracing situation, a different possible world.

[12] The term is from the work mentioned in fn. 23. It is, of course, a metaphor; in particular, Logical Space is not a spatiotemporal region.

Possible worlds thus are situations that are *maximal* in that they are not themselves parts of larger spatiotemporal regions.

As we have seen, actual situations, i.e., segments of reality, can stand in a variety of relations to one another: one may be included in the other or temporally precede it, they may spatially overlap or be far distant from each other, etc. The same is true of *counterfactual* situations, i.e., segments of a non-actual world: every possible world consists of an enormous number of situational parts, which stand in various spatiotemporal relations to one another. And every possible situation, i.e., every element of Logical Space, is part of one possible world and thus part of a spatiotemporal web of relations. Any one of these situations is a quite concrete, specific spatiotemporal segment. Just like the actual situation that I am in consists of innumerable details that I do not have a clue about – from the serial number of the laptop on my left knee via the speed of the train that I am traveling in up to the exact time – so any situation of Logical Space is specified down to the tiniest concrete detail. And since possible worlds are situations too, this is also true of them. A possible world in Logical Space is thus quite different from the world of a novel. For while, e.g., the world of Effi Briest leaves it open whether Innstetten has a mole under his armpit – Fontane did seem to not care about this – the possible worlds of Logical Space are totally specified even with respect to such details. Logical Space therefore hosts many possible worlds that match Fontane's novel, many Effi-Briest-worlds, which only differ from each other in details that have been left open by the novel; and none of these worlds could claim to be *the* world of Effi Briest. Only in their totality do they correspond to the content of the novel. Thus viewed, the *world of a novel* in the everyday sense is a *proposition* in the sense of Logical Space.

We will not impose any limits to the diversity of Logical Space. No conceivable situation, be it ever so outlandish, is going to be excluded from it. We thereby make sure that the contents of arbitrary sentence can be distinguished by the Principal Principle of Semantics, as long it is conceivable tat one of them is true and the other is not. But despite the diversity of Logical Space, a possible situation that separates two given sentences cannot always be found. Thus, e.g., the following two sentences are true of the very same situations – for if one is true, the other one cannot really be false:

(16) **Ein Esel wird von einem Bauern gekauft.**
[≈ A donkey is bought by a farmer.]

(17) **Ein Bauer kauft einen Esel.**
[≈ A farmer buys a donkey.]

According to the Principal Principle of Semantics (16) and (17) thus coincide in context – which is good. For the two sentences obviously do say the same thing.

1.5 Sense Relations in Logical Space

To the extent that content corresponds to meaning, the equality in content observed for (16) and (17) corresponds to a sense relation mentioned in Section 0.2 – synonymy. In this section we will look at further sense relations that may hold between (declarative) sentences. The focus will be on the question of how these sense relations among sentence are reflected in the propositions they express.

Let us start with a case of incompatibility. We had only defined this sense relation for nouns, but it naturally carries over to sentences. For just like **Gedanke** [≈ thought] and **Buch** [≈ book] can never relate to the same object – no thought is a book and vice-versa – so (17) and (18) can never apply to the same situation:

(18) **Niemand kauft ein Tier.**
[≈ Nobody is buying an animal]

Denoting the proposition expressed by a sentence S by '$\|S\|$' – as we will from now on do –, there cannot be any situation s for which it holds that: $s \in \|$**Niemand kauft ein Tier**$\|$ and $s \in \|$**Ein Bauer kauft einen Esel**$\|$. For if s were such a situation, then in s – since (17) is true – there would have to be some farmer who buys a donkey; but then in this situation someone (viz., said farmer) buys an animal (viz., said donkey), whereby (18) would be false as a statement about s – contrary to our assumption. The relation between (17) and (18) can be visualized by a so-called *Venn diagram*, in which sets are represented by regions: In Figure 1.1 the outer rectangle represents

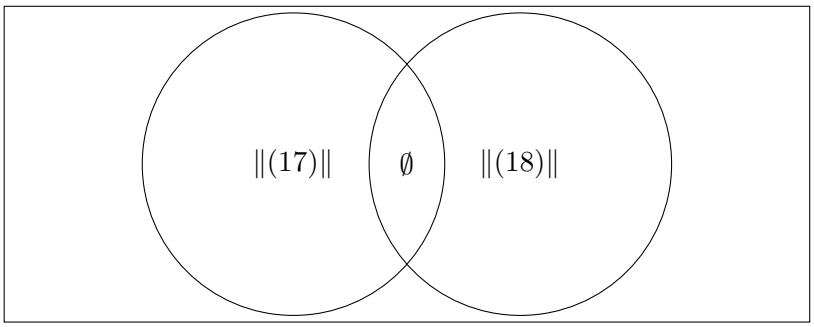


Figure 1.1: *Incompatibility of (17) and (18)*

the Logical Space of all possible situations. The situations of which (17) and (18) are true, are encircled; for brevity, we put their numbers between double lines instead of the sentences themselves, to denote the propositions they express. The region of situations that are covered by both ‖(17)‖ and ‖(18)‖ is the *intersection* of the two propositions – in set-theoretic notation: ‖(17)‖∩‖(18)‖. Here the mere overlap of the circles does not imply that

the two sets have any elements in common. Actually they do not; for as we just saw, there are no such situations of which both (17) and (18) are true. In Figure 1.1 the overlapping area of ‖(17)‖ and ‖(18)‖ has been marked by the symbol 'Ø', which stands for the empty set – the only set without elements.[13]

The incompatibility of (17) and (18) shows in the non-overlap relation between ‖(17)‖ and ‖(18)‖ represented in 1.1: ‖(17)‖∩‖(18)‖ = Ø. In set theory two sets that have no element in common are called *disjoint* (of each other). The example thus shows that the disjointness between propositions comes out as the incompatibility of sentences that express those propositions.

There are more sense relations between sentences that match simple set-theoretic relations between the propositions they express. Let us consider:

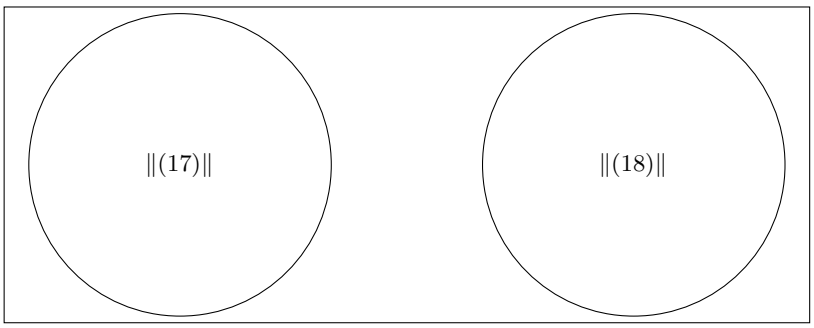
(19) **Niemand kauft eine Kuh.**
[≈ Nobody is buying a cow.]

(18) obviously *implies* (19), i.e., *if* (18) is true, *so* is (19). In relation to Logical Space this means that any possible situation of which (18) is true, is one of which (19) is true. In other words, there is no possible situation to which (18) applies without (19) also applying to it. This state of affairs can again be represented by a Venn diagram: see 1.2[14]

So each element of the proposition ‖(18)‖ is also an element of ‖(19)‖. In set-theoretic terms, ‖(18)‖ thus is a *subset* of ‖(19)‖ – symbolically: ‖(18)‖⊆‖(19)‖. Just like the incompatibility between sentences boils down to a set-theoretic disjunction, implication thus corresponds to subsethood.

If one knows that (18) is true (of a given situation), one knows more than if one only knew that (19) is true (of this situation). In general, if a sentence implies another one, the former contains more, or more detailed, information than the latter. Since implication boils down to subsethood, this

[13] One could also have drawn the two circles so that they do not overlap:



That Ø is the *only* set void of elements is due to the Principle of Extensionality: if L contains no element, then in particular, L and Ø contain the same elements, i.e., L *is* the set Ø.

[14] An alternative representation is:

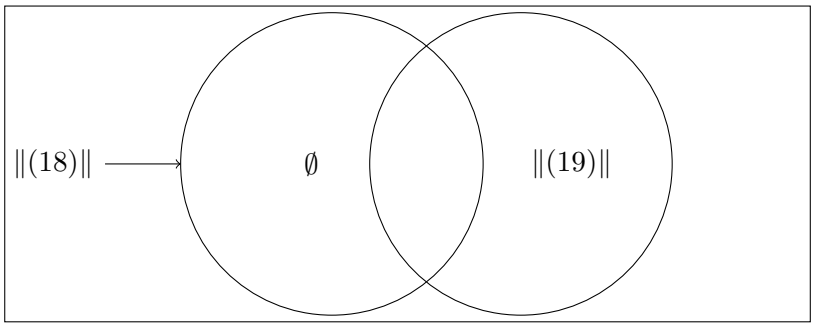
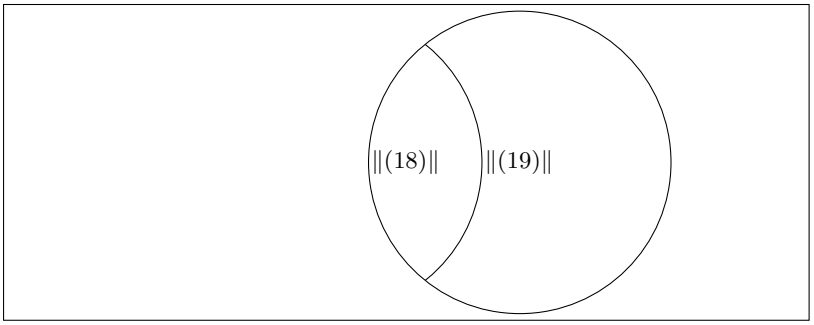


Figure 1.2: *Implication between (18) and (19)*

means: *the more informative* a sentence is, *the smaller* is the proposition it expresses.[15] This seemingly paradox connection becomes clearer once one considers that the more informative sentence *excludes* more situations than the less informative one. Thus, e.g., (19) does not exclude that someone is buying a duck; for (19) contains situations in which Fritz is buying a duck (though nobody is buying a cow). Such situations are excluded by (18), i.e., they are not elements of ‖(18)‖. So ‖(18)‖ is more selective – and hence smaller.

Implication, too, is related to a sense relation that we have already come across in the nominal and adjectival domain: hyponymy. For just like (18) can only apply to a situation to which (19) applies, so the hyponym **Katze** [≈ cat] can only apply to an individual to which the hypernym **Tier** [≈ animal] applies. The exact relation between hyponymy and implication cannot be addressed before Chapter 3 though, when we have the semantic analysis of nouns at our disposal.



[15]The *smaller* set is just the less inclusive one, it does not necessarily have fewer elements: as a rule, propositions are infinite, and so one can be a subset of another one without having fewer elements. This *paradox of infinity* may be elucidated by a mathematical analogy: the set of even numbers {2,4,6, ...} is a subset of the positive natural numbers {1,2,3,4, ...}, but the two sets have equally many elements in that they can be mapped onto each other in a one-one fashion.

1.6 Coordinating Sentences

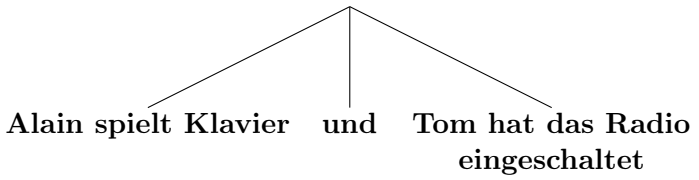
The set-theoretic analysis of sentence contents as propositions sheds a light on the meaning of coordinating conjunctions like **und** [≈ and] and **oder** [≈ or] as they are used in the following sentences:

(20) **Alain spielt Klavier, und Tom hat das Radio eingeschaltet.**
[≈ Alain is playing the piano and Tom has turned the radio on.]

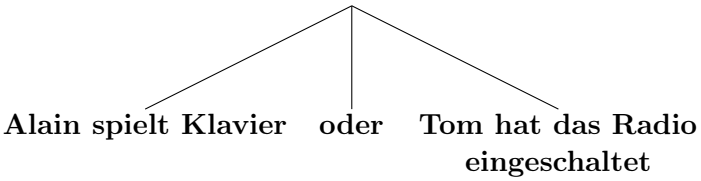
(21) **Alain spielt Klavier, oder Tom hat das Radio eingeschaltet.**
[≈ Alain is playing the piano or Tom has turned the radio on.]

(20) und (21) can be interpreted compositionally. To do so, we will underlay a *ternary* (= three-way) bracketing:

(20′)



(21′)



According to the Principle of Compositionality, the meaning of sentence (20) is obtained by combining the meanings of its immediate parts. According to (20′), (20) has three parts, viz.:

(22) a. **Alain spielt Klavier**
[≈ Alain is playing the piano.]
b. **und**
[≈ and]
c. **Tom hat das Radio eingeschaltet**
[≈ Tom has turned the radio on.]

Two of the three parts of (20) are sentences. Hence their contents, ‖(22a)‖ and ‖(22c)‖, are propositions. ‖(22a)‖ is the set of situations in which Alain is playing the piano, ‖(22c)‖ consists of the situations in which Tom has turned on the radio. Of course, there are also situations in which both is the case; these are obviously just the situations to which (20) applies. As illustrated in figure 1.3, ‖(20)‖ thus is the *intersection* of the propositions expressed by the sentential parts of ‖(20)‖: ‖(20)‖ = ‖(22a)‖∩‖(22c)‖.

Before going into the question of how the content of (20) is determined compositionally, let us first take a look at (21): which situations does the

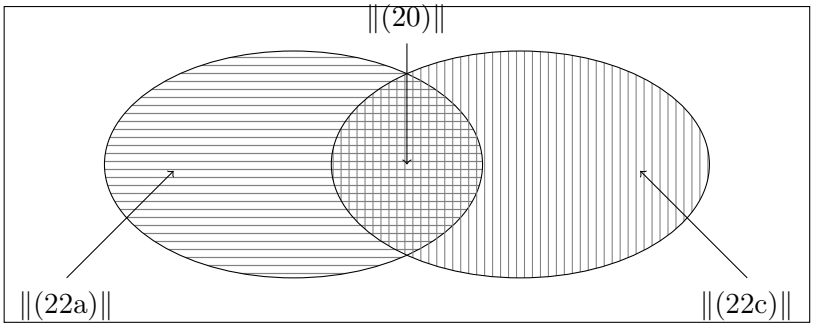


Figure 1.3: ‖(20)‖, ‖(22a)‖ *and* ‖(22c)‖

sentence apply to? First of all, ‖(21)‖ contains all situations in ‖(22a)‖; for as soon as Alain plays the piano, (21) is true. Likewise ‖(21)‖ comprises all situations in ‖(22c)‖; for if Tom has turned the radio on, (21) is true, too. In situations that lie outside both ‖(22a)‖ and ‖(22c)‖, Tom has neither turned on the radio, nor is Alain playing the piano. (21) does not apply to such situations, but it does apply to all others. As a consequence, ‖(21)‖ is the *union* of ‖(22a)‖ and ‖(22c)‖. In set-theoretic notation: ‖(21)‖ = ‖(22a)‖∪‖(22c)‖ – and graphically:

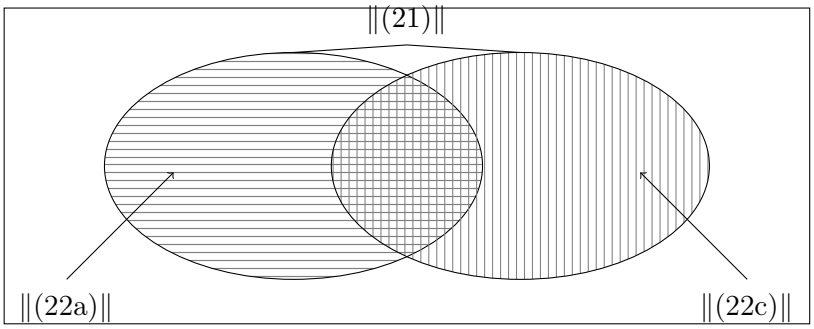


Figure 1.4: ‖(21)‖, ‖(22a)‖ *or* ‖(22c)‖

Following the Principle of Compositionality, ‖(20)‖ must be obtained from ‖(22a)‖, ‖(22c)‖, and the content of **und** [≈ and]; analogously, ‖(21)‖ is obtained from ‖(22a)‖, ‖(22c)‖, and the content of **oder** [≈ or]. Abstracting from the accidental feature of the example, the following interpretation of **und** and **oder**-connections emerges – or *conjunctions* and *disjunctions*, as they are called in logical semantics:[16]

[16]The terms *conjunction* and *disjunction* are also used for the words **und** [≈ and] and **oder** [≈ or] themselves, as well as for the meanings of these words. The terminology is somewhat unfortunate in that a *conjunction* in the syntactic sense is a particular category (part of speech), to which both **und** [≈ and] and **oder** [≈ or] belong. But the

(23) *Semantics of conjunction*
If S and S' are declarative sentences, the following holds:
$\|S$ **und** $S'\| = \|S\| \cap \|S'\|$.

(24) *Semantics of disjunction*
If S and S' are declarative sentences, the following holds:
$\|S$ **oder** $S'\| = \|S\| \cup \|S'\|$

The general format of (23) and (24) is reminiscent of rule (15) [*Evaluation of Complex Terms*] from the preceding chapter, where [juxtaposition and] superscripting was interpreted as expressing an arithmetical operation – exponentiation. In a similar vein, (23) and (24) interpret conjunction and disjunction – the combination with **und** or **oder** – by set-theoretic operations, viz. intersection ($\cap$) and union ($\cup$). However, the analogy is false given that superscripting is doubtlessly a kind of combination of terms and thus a syntactic construction of the language of mathematical formulae. But conjunction and disjunction are no constructions on their own. They are rather applications of the same construction called *coordination*, which is the combination of two sentences by a coordinating conjunction. If conjunction and disjunction are to be interpreted as special cases of coordination, then the words **und** and **oder**, too, need to have their own lexical content, which can then be combined with the contents of the sentences conjoined. It is now natural to regard the set-theoretic operations themselves as these lexical contents:[17]

(25) *Lexical Semantics of Conjunction and Disjunction*
$\|$**und**$\| = \cap$; $\|$**oder**$\| = \cup$.

Unlike (23) and (24) in its general format, (25) is like the *Evaluation of Simple Terms* (11) from the preceding chapter, putting, e.g., $[\![\mathbf{5}]\!] = 5$. (25), too, lists the values for certain simple, lexical expressions. But then (25) appears anything but trivial; for that the word **und** [$\approx$ and] expresses intersection of propositions is not part of general school education (unfortunately!). In what follows we will be assuming that (25) is part of the semantics of German.

Instead of treating conjunction and disjunction as distinct constructions, as in (23) and (24), it appears more adequate to assume a single construction and trace back the difference to a difference in word content. In (25) it is laid down what the latter is. But what is still missing indeed, is a reduction of the content of sentences like (20) and (21) to the meanings of the words **und** [$\approx$ and] and **oder** [$\approx$ or]. This is done by the following general rule:

(26) *Semantics of sentence coordination*
If S and S' are (declarative) sentences and K is a coordinating con-

context makes it usually clear what is meant.

[17] In (i) the symbols '$\cap$' and '$\cup$' must be interpreted as operations on propositions, not on sets in general; for in the latter sense intersection and union are no set-theoretic objects.

junction, then the following holds:
$\|S\ K\ S'\| = \|S\|\ \|K\|\ \|S'\|$.

The notation used in the above equation may be irritating. But then it only exploits the fact that the content of a coordinating conjunction is a set-theoretic operation and the schematic '$\|K\|$' in each case can be replaced by (the symbol for) such an operation – i.e., by '$\cap$', '$\cup$', etc. As a case in point, for the case that K = **oder**, (26) ultimately yields the same result as in (24):

(27)	$\|S\ K\ S'\|$	
=	$\|S\|\ \|K\|\ \|S'\|$	following (26)
=	$\|S\|\ \|$**oder**$\|\ \|S'\|$	if K=**oder**
=	$\|S\| \cup \|S'\|$	due to (25)

In (24) we had thought of, and interpreted, disjunction as a construction of its own; the meaning of this construction was a semantic operation, to wit, set union (restricted to propositions), which takes two propositions p and q at a time and 'merges' them into a third one, the union $p \cup q$. In (27), however, we had been assuming that disjunction is a special case of sentence coordination; the content of a disjunction like (21) then results from an interaction of the word content of **oder** [≈ or] and the interpretation of sentence coordination as given in (26). The latter is a semantic operation that 'merges' three contents at a time – two propositions p and q and a set-theoretic operation O – into a single proposition $p\ O\ q$. In semantics, this abstract operation is known as *functional application*; for, mathematically speaking, it has a *function* (the set-theoretic operation expressed by the conjunction, in this case) *apply* to its *arguments* (the propositions expressed by the resulting coordinated sentence). We will come across functional application again and again during the course; the concept of a function itself will be addressed in the following section.

Let us end the section with a digression on the pragmatics of disjunction. According to figure 1.4 – and the interpretation of **oder** [≈ or] given in (25) – (27) also applies to situations in which Alain is hammering on the piano and Tom lets his radio drone. This may appear dubious at first sight: doesn't (21) say that *either* (22a) *or* (22c) is true, but *not both*? Not necessarily. If I use (21) as an explanation and an apology for the high level of noise toward my guest, the latter can hardly blame me for lying or being wrong should it turn out that the acoustic disturbance was due to my two sons' joint activities. The proposition expressed by my utterance thus covers *all* situations in ‖(22a)‖ and ‖(22c)‖, including those in the intersection. The opposite impression may at best arise if the latter situations are somewhat less likely for speaker and hearer than the other possibilities in‖(21)‖. But what speaker and hearer are thinking about while communicating, is not necessarily relevant to linguistic meaning; we had already mentioned that in

Section 1.1.

In many cases, though, a disjunction actually excludes the corresponding conjunction. Someone who utters a sentence like (28), thereby does not mean that (29) might be the case:[18]

(28) **Fritz ist in der Uni oder [er ist] noch unterwegs [zur Uni].**
[≈ Fritz is a the university or [he is] still on his way [to university]]

(29) **Fritz ist in der Uni und [er ist] noch unterwegs [zur Uni].**
[≈ Fritz is a the university and [he is] still on his way [to university]]

This is no reason to believe that (28) is a different, 'exclusive' type of disjunction though. For the the two clauses connected by **oder** [≈ or] express disjoint propositions in the first place: the entire Logical Space does not contain a single situation in which Fritz is both at the university and on his way to university – for the latter presupposes that he has not (yet) arrived at his goal. It is precisely for this reason, however, that (28) can be interpreted as union in the style of (27); though the intersection of the propositions expressed by the two clauses is covered by $\|(28)\|$, this intersection is empty anyway. The fact that an utterance of (28) excludes the truth of (29), then, has nothing to do with the meaning or the content of **oder** [≈ or].

Not in every case where a disjunction excludes the truth of the corresponding conjunction, is this due to the impossibility of the latter; it suffices that the *falsity* of the conjunction may be assumed as *known*. Thus (30) *per se* does not preclude that Eric brings two kids along; but if speaker and hearer know that Eric has only one child, the second option is out for both:

(30) **Eric bringt heute seinen Sohn oder [er bringt heute] seine Tochter mit.**
[≈ Today Eric is bringing his son along or [he is bringing] his daughter [along]]

(31) **Eric bringt heute seinen Sohn und [er bringt heute] seine Tochter mit.**
[≈ Today Eric is bringing his son along and [he is bringing] his daughter [along]]

In this case, then, the truth of the conjunction (31) is not excluded by Logical Space but by the *conversational background*, the knowledge presupposed as common to speaker and hearer for communicative purposes. The exact relation between the conversational background and the proposition expressed as well as the interaction between them is studied in pragmatics.[19] At least

[18] These sentences are more idiomatic if the bracketed parts are omitted, but then the fact they are *sentence* coordinations (in Logical Form) gets clouded; we will return to these matters in Chapter 6.

[19] In this connection the fact will have to be addressed that in the circumstances described, (30) is only adequate if the gender of Eric's (only) child is unknown to the speaker.

for cases like (30) the impression of an exclusive reading can be dispelled by relatively easy means. Whether this is always possible is an open question though. If not, two homophonous conjunctions would have to underly the surface form **oder** [≈ or] and (25) would have to be adapted accordingly. We shun this complication and assume for the following that **oder** [≈ or] is neither homonymous nor polysemous and (25) fully captures the content of its use as a sentence coordinator.

1.7 Extension and Intension

The chapter ends with a formal variant of the notion of a proposition that the rest of these notes will be based on. Other than the set-based concept introduced above, this variant approaches the content of a sentence on the background of Logical Space, which is partitioned into a **YES**-area (= the situations to which the sentence applies) and a **NO**-area (= those to which it doesn't apply):

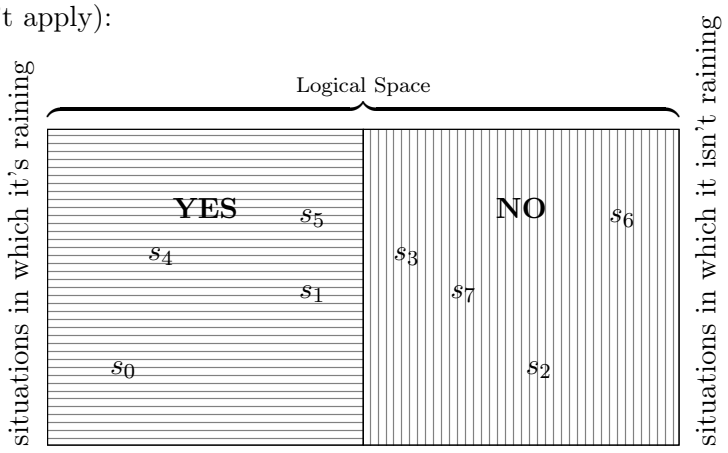


Figure 1.5: *The content of a sentence as a bipartition of Logical Space*

Using the (random) example **Es regnet** [≈ It is raining] as a model, figure 1.5 shows how the content of a sentence divides Logical Space, by *assessing* each (possible) situation according to whether the sentence is true of it. Using, as is customary in logic,[20] the *truth values* 1 and 0 as indicators for the correctness (**YES**) and incorrectness (**NO**) of a sentence, the partition of Logical Space represented in figure 1.5 can also be written in the form of a table:

[20] ... since George Boole's *The Mathematical Analysis of Logic* (Cambridge 1847); the term 'truth value' [German 'Wahrheitswert'] goes back to Frege.

Situation	Truth Value
s_0	1
s_1	1
s_2	0
s_3	0
s_4	1
s_5	1
s_6	0
s_7	0
...	

Table 1.1: *Table representing a bipartition of Logical Space*

In Table 1.1 the individual situations are assigned their truth value according to the bipartition of Logical Space in figure 1.5. Mathematically speaking, this bipartition thus comes out as a *function*. The situations listed in the left column of the table, those to which the function assigns something, are called the *arguments* of the function; we are assuming that any possible situation of Logical Space can serve as an argument of the function represented in Table 1.1. The object the function assigns to an argument is called the *[functional] value* for this argument. Each argument may only be assigned one value; this is the main characteristic of a function. If f is a function and x an argument, its functional value is written: '$f(x)$'. The set of all arguments of a function f is also called its *domain* – for short: $dom(f)$. The set of all values is also called the *range* – for short: $rge(f)$. These mathematical terms are quite general and will from now on be used whenever we are dealing with functions.

Functions may be construed as particular sets whose elements are so-called *ordered pairs*. As the name suggests, an ordered pair always consists of two objects, its first and its second *component*. We will write '$(x,\ y)$' for a pair with first component x and second component y.[21] It is important to notice that the pair $(x,\ y)$ differs from the set $\{x,\ y\}$ in that the *order* of the components is vital: if x and y are two distinct objects, then $(x,\ y)$ and $(y,\ x)$ are also distinct (whereas $\{x,\ y\} = \{y,\ x\}$, as we already saw in Section 1.2 in connection with the Extensionality Principle). And what is more: a pair $(x,\ y)$ uniquely determines its first and second components in that there are not supposed to be any other objects x' and y', that are the first and second components of the same pair. On the other and, one can also construct the pair of an object x with itself – and this pair $(x,\ x)$ then neither coincides with x itself nor with the singleton set $\{x\}$ (whereas $\{x,\ x\} = \{x\}$, as we had seen too). Given these difference, it is quite surprising that it is in principle possible to reduce the concept of a pair to that of a set. Since this

[21] Another common notation for ordered pairs $(x,\ y)$ is by angular brackets: '$\langle x,\ y\rangle$'.

reduction does not play any rôle here, we are hiding it in an exercise though. In what follows we will simply assume that pairs are asymmetric objects in that swapping the first and second components results in a different pair – unless the two components are identical.

In mathematics functions are commonly identified with sets of ordered pairs whose first components are the arguments and whose second components are the corresponding values. Thus the function f is the set of all pairs $(x, f(x))$, where x is any argument. But beware: not every set of ordered pairs is a function![22] Rather, it has to meet a *uniqueness condition* (aka *functionality*) to the effect that no argument must have more than one value: if a pair (x, y) is in the function f (and thus $f(x) = y$), then the same function must not contain another pair (x, z), for which it holds that $y \neq z$; otherwise we would have: $y = f(x) = z$, contradicting the assumption that $y \neq z$. In the case of the function represented in Table 1.1 this means that the content of the sentence **Es regnet** [≈ It is raining] is not both true and false of any situation, which is something we are indeed taking for granted, following a venerable principle of logic.[23]

Clearly, then, the bipartition of Logical Space by the content of a sentence can be represented by a function that assigns to any situation a truth value, i.e., 0 or 1. Function and sentence content correspond to each other in that one can be derived from the other. The function corresponding to the content of a sentence assigns the truth value 1 to each element of the content, and 0 to all another situations; seen as a set, this function thus consists of all pairs $(s, 1)$ where s is an element of the content, plus all pairs $(s, 0)$ where s is not an element of the content (but still a possible situation from Logical Space). Inversely, one may start with a bipartition of Logical

[22] Sets of ordered pairs in general are called *(binary) relations*; hence functions are special relations.

[23] The *Law of Non-Contradiction*, which has already been formulated in classic antiquity, says that a sentence and its negation – **Es regnet** [≈ It is raining] and **Es regnet nicht** [≈ It isn't raining], in the case at hand – cannot be true at the same time, i.e., relating to the same situation. Concerning situations in which it is drizzling very lightly, one may be tempted to say that it is on the one hand raining in them, and on the other hand not really – depending on the standard for 'proper' rain. This illustrates the general semantic-pragmatic phenomenon of *vagueness*, which we are ignoring throughout these notes. Instead we are assuming that vague concepts are to be made precise for purposes of semantic analysis, e.g., by introducing artificial imaginary standards. More about the topic of vagueness can be found in Chapter 2 of Mark Sainsbury's *Paradoxes* (2nd ed.,1995). – A further reason for violating the Law of Non-Contradiction may be seen in the size of certain situations: in World War II it sometimes rained in some places, and sometimes it didn't rain in some places. Does the sentence **Es regnet** [≈ It is raining] as relating to the war situations have two truth values, then? No, because we may assume that the truth value is 1 because it had been raining in (at least) one of its situational parts after all. This assumption ought to fall out of a lexical analysis of **Es regnet** [≈ It is raining], expressing a *persistent* proposition, one whose truth carries over from small situations to larger ones containing them. We must leave such finesses of lexical analysis out of account here.

Space represented by a function and form a corresponding proposition by including all (and only those) situations to which the function assigns the value 1. This correspondence between sets and functions whose arguments are assigned truth values is quite general: if U ($\neq \emptyset$) is some (non-empty) set, then the subsets of U match the functions whose domain is U and whose values are always truth values.[24] These functions are said to *characterise* the corresponding sets and are thus called *characteristic* functions:

D1.5 The *characteristic function* of a set M (relative to a set U) is a function f with domain U such that for any $x \in U$ it holds: $f(x) = 1$ iff $x \in M$, and $f(x) = 0$ iff $x \notin M$.

In the case at hand, U is Logical Space, and the sets characterized are the propositions. Further examples of characteristic functions will be encountered later in the course. Their domain will not always coincide with Logical Space.

The characteristic function of the content of a sentence S is called the *Intension* of S. From now on we will write it as '$[\![S]\!]$'. The truth value the intension assigns to a situation s is called the *extension* of the sentence S (at the situation s). Traditionally – and here, too – it is written with the argument as a superscript: '$[\![S]\!]^s$'.

D1.6 The *intension* $[\![S]\!]$ of a (declarative) sentence S is the characteristic function of its content (relative to Logical Space), i.e., that function f with domain Logical Space and such that for every situation s the following holds:
$f(s) = \left\{ \begin{array}{ll} 1, & \text{if } s \in \|S\| \\ 0, & \text{if } s \notin \|S\| \end{array} \right\}$

D1.7 The *extension* $[\![S]\!]^s$ of a (declarative) sentence S at a given situation s is the value its intension assigns to s: $[\![S]\!]^s = [\![S]\!](s)$.

Given this notation, a simple connection between contents of sentences and their extensions can be formulated that will be of importance later:[25]

(32) For any situation s of Logical Space and any sentence S the following holds:
a. $s \in \|S\|$ iff $[\![S]\!]^s = 1$.
b. $s \notin \|S\|$ iff $[\![S]\!]^s = 0$.

The conceptual pair *extension/intension* will accompany us throughout the

[24] This means that the range is a subset of $\{0, 1\}$. It does not have to coincide with this set, because the function might always assign the same value; in this case its range only contains one element of $\{0, 1\}$.

[25] The second observation in (32) of course follows from the assumption that $[\![S]\!]$ is a characteristic function.

course.[26] More specifically, we will try to define both extensions and intensions for all kinds of linguistic expressions – nouns, verbs, articles, prepositions, ... Intensions will more or less correspond to the literal meanings of the expressions. The extension, on the other hand, always connects the intension with a situation: the extension indicates what the expression relates to in a given situation, it determines its *reference*; thus extensions are always situation-dependent. In the case of sentences – the only expressions for which we have defined extensions so far – this connection is admittedly somewhat indirect. If a sentence S is true of a (possible) situation s, one may take s itself to be the object to which S refers. If, however, S does not apply to a situation s, the sentence does not refer to anything in s. In this sense the extension of a sentence at a situation s always indicates what S refers to: if $[\![S]\!]^s = 1$, S refers to s; if, on the other hand, $[\![S]\!]^s = 0$, S has no (actual) referent.

As the extensions of sentences, truth values have a surprising property that will help us to transfer the notion of extension to the clausal conjunctions **und** [≈ and] and **oder** [≈ or] considered in the preceding section. Let us first observe that the extension of a coordination is determined by the extensions of the parts coordinated:

(33)	$[\![S$ **und** $S']\!]^s = 1$	
iff	$s \in \|S$ **und** $S'\|$	by (32)
iff	$s \in \|S\| \cap \|S'\|$	by (23)
iff	$s \in \|S\|$ and $s \in \|S'\|$	by def. of '$\cap$'
iff	$[\![S]\!]^s = 1$ and $[\![S']\!]^s = 1$	by (32)
(34)	$[\![S$ **oder** $S']\!]^s = 1$	
iff	$s \in \|S$ **oder** $S'\|$	by (32)
iff	$s \in \|S\| \cup \|S'\|$	by (24)
iff	$s \in \|S\|$ or $s \in \|S'\|$	by def. of '$\cup$'
iff	$[\![S]\!]^s = 1$ or $[\![S']\!]^s = 1$	by (32)

In (33) and (34) the truth value of a coordination is traced back to the truth values of the coordinated clauses. The extensions of coordinations thus turn out to be compositional. This becomes even clearer if one takes into account that truth values are ultimately numbers and one may in principle calculate with them.[27] As one readily verifies, the conditions given in (33) and (34) amount to simple arithmetical combinations. The conditions given in (33) and (34) thus turn out to be parallel to the compositional accounts of the

[26] The method of extension and intension, originating with Rudolf Carnap's book *Meaning and Necessity* (1947), is a modification of Frege's approach in *Über Sinn und Bedeutung* (1892) and owes its current form to American logician Richard Montague's work mentioned in footnote 15 of Chapter 0.

[27] There are, by the way, logical and mathematical reasons for identifying the extensions of sentences with the numbers 0 and 1, which we cannot go into here now.

contents of coordination given in (23) and (24) above:

(35) a. $\|S \textbf{ und } S'\| = \|S\| \cap \|S'\|$ cf. (23)
b. $[\![S \textbf{ und } S']\!]^s = [\![S]\!]^s \times [\![S']\!]^s$ (33) + mental arithmetic

(36) a. $\|S \textbf{ oder } S'\| = \|S\| \cup \|S'\|$ cf. (24)
b. $[\![S \textbf{ oder } S']\!]^s = [\![S]\!]^s + \|S'\| - [\![S]\!]^s \times [\![S']\!]^s$
(34) + mental arithmetic

Multiplication is to the extensions of conjunctions what intersection is to their contents; and the difference of product and sum is to the extensions of disjunction what union is to their contents. It is thus natural to carry over the general compositional analysis (26) of coordinations from contents to extensions:

(37) a. $\|S\ K\ S'\| = \|S\|\ \|K\|\ \|S'\|$ cf. (26)
b. $[\![S\ K\ S']\!]^s = [\![S]\!]^s\ [\![K]\!]^s\ [\![S']\!]^s$

(37b) can be read in full analogy to (37a): the extensions of the two clauses S and S' are combined by the extension of the conjunction K and the result is the extension of the coordination, i.e., the truth value of the entire sentence. (37b) presupposes that the extension of a coordinating conjunction combines two truth value into a truth value again – just like union and intersection have a proposition emerge from two propositions. The extension of **und** [≈ and] would thus have to be the multiplication of truth values; and the extension of **oder** [≈ or] would be the operation described in (36b). Both may be represented in tabular form – by so-called *truth tables*:

left truth value	right truth value	**result**
1	1	**1**
1	0	**0**
0	1	**0**
0	0	**0**

Table 1.2: *Tabular representation of the extension of **und** [≈ and] (at an arbitrary situation)*

left truth value	right truth value	**result**
1	1	**1**
1	0	**1**
0	1	**1**
0	0	**0**

Table 1.3: *Tabular representation of the extension of **oder** [≈ or] (at an arbitrary situation)*

It should be noted that Tables 1.2 and 1.3 give the extensions of **und** [≈ and] and **oder** [≈ or] for an arbitrary situation s. This brings out a characteristic of so-called 'logical' words, viz. that they do not really relate to a situation: their contribution is always the same, no matter to which situation they may relate. But they can, of course, help a sentence in which they occur refer to a situation; for the latter is reflected in its – usually situation-dependent – truth value determined by the relevant truth table.

The intensions of the conjunctions **und** [≈ and] and **oder** [≈ or] are defined in analogy to sentence intensions, viz., as functions that assign to each situation s the corresponding extension at s. So the domain of $[\![\textbf{und}]\!]$ and $[\![\textbf{oder}]\!]$ is Logical Space – which is so for intensions in general, as we will see later. Since the extension of **und** [≈ and] is always the function described in Table 1.2, the range of $[\![\textbf{und}]\!]$ only contains one element, viz. precisely this function; and the same holds for $[\![\textbf{oder}]\!]$. The two extensions are thus *constant* functions, meaning that they assign the same value to every argument in their domain:

(38) a. $[\![\textbf{und}]\!]$ = that function f with domain LS (= Logical Space) such that, for any $s \in LS$ it holds: $f(s) = [\![\textbf{und}]\!]^s$.
b. $[\![\textbf{oder}]\!]$ =that function f with domain LS (= Logical Space) such that, for any $s \in LS$ it holds: $f(s) = [\![\textbf{oder}]\!]^s$.

In a way, (38) defines the intensions of the conjunctions **und** [≈ and] and **oder** [≈ or] by reference to their extensions (given in tables 1.2 and 1.3). We will do the same in the future: no matter for which expression, we will first determine their extensions (as depending on a given situation) and then characterise the intension as that function that assigns to each situation s the extension at s. The extensions themselves will be determined independently, largely by compositionality considerations. In the next chapter we will begin to illustrate this general strategy by very simple examples.

1.8 Exercises for Chapter 1

A1 Explain how one can show that two sentences are not synonymous by appealing to the Principal Principle of Semantics. Pick your own example for illustration.

A2 What are the contents of the following sentences?

I **Keiner lacht.**
[≈ Nobody is laughing.]

II **Politik ist Politik.**
[≈ Politics is politics.]

III **Politik ist nicht Politik.**
[≈ Politics is not politics.]

IV **Tim Fernando kam am 23.10.2008 zu Besuch.**
[≈ Tim Fernando came visiting on October 23, 2008.]

V **Es gibt keine Universität in Frankfurt am Main.**
[≈ There is no university in Frankfurt/Main.]

A3 Use your own examples to discuss what speaks against the assumption that there are sentences that only apply to one or two situations.

A4 The following sentences obviously state the exact opposite of each other; one *negates* the other:

I **Ein Bauer kauft einen Esel.**
[≈ A farmer is buying a donkey.]

II **Kein Bauer kauft einen Esel.**
[≈ No farmer is buying a donkey.]

(a) How does the sense relation of negation differ from incompatibility?

(b) Use a Venn diagram to represent the set-theoretic relation between I and II.

A5 Illustrate each of the following sense relations by two sentences of your own and describe how their contents relate to each other.

I Incompatibility

II Synonymy

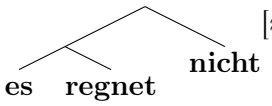
III Logical implication

A6 Like the Logical Space LR of all possible situations, the empty set $\emptyset$ is a proposition: both sets exclusively consist of (possible) situations.

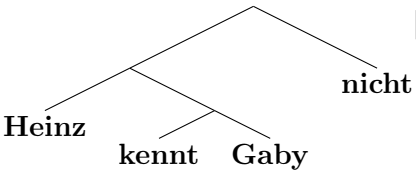
(a) Find a sentence that expresses $\emptyset$ (as a proposition).

(b) Find a sentence that expresses LR (as a proposition).

A7 Assume – for the sake of this exercise only – that the negation **nicht** [≈ not] may modify entire sentences, as in:



[≈ It is not raining.]



[Heinz does not know Gaby.] , etc.

Use Venn diagrams to describe the meaning of **nicht** [≈ not]. Try to make your presentation as parallel as possible to the above analysis of coordinating conjunctions.

A8 Why isn't every set of ordered pairs a function (example)?

A9 Briefly explain the relation between the contents and the intensions of sentences in your own words.

A10 Show that the intersection of two distinct characteristic functions (relative to a given set) is not a characteristic function (relative to that set).

A11 If x and y are arbitrary (not necessarily distinct) objects, the *Kuratowski pair of x and y*[28] is the set $\{\{x,\ y\},\{x\}\}$. Show that this set meets anything one might expect from a pair (x,y):

(a) if $\{\{x,y\},\{x\}\} = \{\{x',y'\},\ \{x'\}\}$, then $x = x'$ and $y = y'$;

(b) if $x \neq y$, then $\{\{x,y\},\{x\}\} \neq \{\{y,x\},\{y\}\}$;

(c) $\{\{x,x\},\{x\}\} \neq \{x\}$.

Hint: Apart from the Extensionality Principle, (c) also relies on the *Foundation Principle*, according to which (among other things) no set can be its own element.

[28] named after its inventor, the Polish logician Kazimierz Kuratowski [1896–1980]

Chapter 2

Predication and Abstraction

Having become sufficiently clear on what sentences as a whole mean, we now turn to their parts. As announced above, we will use a top-down strategy, concluding from the meanings of entire sentences to the meanings of their *immediate* parts, then to the meanings of the immediate parts of the latter etc. – until we ultimately arrive at the word meanings. To this end, we will extend the concepts introduced in connection with sentence meanings to arbitrary parts and equate meanings with *intensions* – functions that for a given situation determine an *extension*, which in turn supplies the reference of the expression for that situation. Moreover, we will (until further notice) assume that these extensions behave *compositionally* so that the extensions of complex expressions can be determined from those of their immediate parts; in the preceding chapter, we had already come across this procedure by way of an example: the interpretation of sentence coordination. In the chapter at hand, too, we will first cut up sentences into their immediate parts and then define extensions for them that can be combined so as to result in the truth values of the sentences.

We will start from the sentence extensions introduced in the preceding chapter – the truth values – which we will try to determine from the parts of the sentence. Whatever the extensions of the parts of a given complex expression may be, the Principle of Compositionality demands that they have to 'add up' to the extension of the entire expression. As we will soon see, this simple pre-theoretic observation will become highly fruitful, because in many cases it will allow extensions to be defined by *taking differences*. In Section 2.2 we will see how this works in detail. Before that, we still need to find a suitable starting point. This will be the simplest construction from a semantic point of view (apart of the above sentence coordination): *(elementary) predication*.

2.1 Proper Names

We start with the following, maximally simple example (by semantic standards):

(1) **Olaf hat Husten.**
[≈ Olaf has a cough.]

What makes sentence (1) so simple is the fact that one of its immediate parts – the subject – is a proper name. Such sentences will be called (subject) *predications*.[1] Complex sentences like (2) are predications, too. (3) and (4), on the other hand, are not predications even though they contain proper names, but not as *immediate* parts.

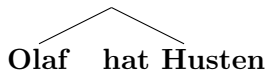
(2) **Erna trifft einen Bekannten, den sie lange nicht mehr gesehen hat.**
[≈ Erna is meeting an old acquaintance that she has not seen for a long time.]

(3) **Jeder kennt Otto.**
[≈ Everyone knows Otto.]

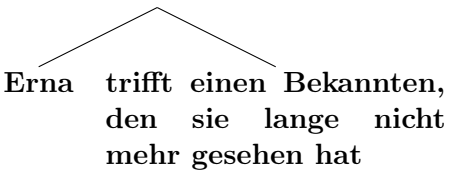
(4) **Heinz schläft, und Gaby arbeitet.**
[≈ Heinz is sleeping and Gaby is working.]

Dissecting (1) and (2) into their immediate parts is straightforward:

(1′)



(2′)



From a semantic point of view, (1) and (2) are only slightly different: they are both predications, i.e., they consist of a predicate and a proper name as their subject. Following the Principle of Compositionality, the sentence extension must be obtained from the extensions of subject and predicate in the same way in both cases. As long as we are only interested in predication, nothing is lost if we concentrate on the shorter sentence (1). The analysis of (2) proceeds in a fully analogous fashion; sentences like (3), on the other hand, will only be scrutinized in the next chapter.

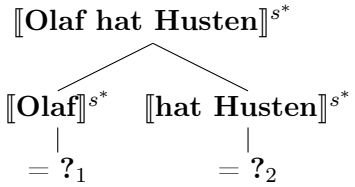
According to the Principle of Compositionality, some combination of extensions that corresponds to predication (as a construction) needs to 'merge' the following two ingredients into the sentence extension – i.e., truth value of (1) – in an arbitrary situation s^*:

[1] The term comes from logic, where it is used in a slightly wider sense and defined in purely semantic terms.

- $[\![\textbf{Olaf}]\!]^{s^*}$, the extension of the subject **Olaf** (in s^*)
- $[\![\textbf{hat Husten}]\!]^{s^*}$, the extension of the predicate **hat Husten** (in s^*)

Though we know what the result of the interaction of these ingredients is, viz. $[\![(1)]\!]^{s^*}$,[2] but the ingredients themselves are still as unknown to us as the semantic operation that merges them. The latter will more or less fall out once we have found the ingredients to be combined. We may illustrate the deplorable state of our current knowledge of the compositional interpretation of (1) as follows:

(1″)



We will eliminate both question marks in the course of this chapter, stating with the left one, $?_1$. Once the first question mark has been removed, we will introduce a method that allows to systematically derive the predicate extension, which will happen in the next section.

A simple plausibility consideration immediately leads to a 'minimalist' answer to the question of what the extension of **Olaf** is. Since the extension ought to be that object that corresponds to the reference of an expression, it is natural to identify the extension of a name with its *bearer*:

(5) a. $[\![\textbf{Olaf}]\!]^{s^*}$ = Olaf;
b. $[\![\textbf{Maria}]\!]^{s^*}$ = Maria;
c. $[\![\textbf{Berlin}]\!]^{s^*}$ = Berlin;
etc.

In (5), s^* is an arbitrary situation in Logical Space, on which the above extensions of proper names do not really depend though. In this respect proper names are thus like the logical words **und** [≈ and] and **oder** [≈ or] and different from most sentences whose truth values usually depend on the situation they are taken to describe. This situational independence assumed in (5) may appear problematic at first since in different situations different individuals (persons, cities, ...) may be the bearers of a given proper name. But this impression is misleading. For on the one hand, we assume that any name can only have one bearer. Of course, there a lot persons called **Hans** or **Hans Müller**, and of course there is more than one

[2] More precisely, we know that the extension of (1) is the truth value of (1) in s^*; this knowledge suffices for the following semantic considerations. Just which truth value it is – 1 or 0 – is only known to us if we are sufficiently familiar with the details of the situation s^*, which in turn depends on how s^* is given to us – by description, memory, observation etc.

place called **Berlin** or **Frankfurt**. But for these case we assume that the names are ambiguous forms (homonyms) that would strictly be differentiated – by subscripts, say. In that sense the name **Frankfurt** in the reading $\textbf{Frankfurt}_{\text{Main}}$ has only one bearer per situation. On the other hand, the fact that there are counterfactual situations s^* in which, say, my sons' Christian names had been swapped, is irrelevant for determining the extension of the two names. For the extension is that object to which we *actually* refer by the name when talking of said situation s^*. What language is spoken in s^*, or if any language is spoken at all, plays no rôle: if, say, in such a counterfactual s^*, Alain is calling his brother, then he could do so by utterance the phonetic form **Alain!** and Tom may respond with an unnerved **Was ist denn, Tom?** [≈ What's up, Tom?]. The underlined occurrences of the two names show that we are referring to the two by their actual names – despite the name swap in s^*; accordingly, it is their actual bearers that constitute the extensions of the (actual) names in s^*.[3]

In (5) we have listed the *extensions* of some sample proper names *NN* for any given situation s^*: *NN* = **Olaf**, *NN* = **Maria**, *NN* = **Berlin**, etc. As is the case with every expression, the *intension* of the name now assigns to each situation the corresponding extension:

Situation	*Extension*
s_0	$[\![NN]\!]^{s_0}$
s_1	$[\![NN]\!]^{s_1}$
s_2	$[\![NN]\!]^{s_2}$
...	...

Table 2.1: *Tabular representation of the intension* $[\![NN]\!]$ *of a proper name NN*

We just saw that extension of a proper name is independent of the situation considered – which is why the values in the right column of Table 2.1 are all the same: $[\![NN]\!]^{s_0} = [\![NN]\!]^{s_1} = [\![NN]\!]^{s_2} = \ldots$. Represented as a table, the intension of the name **Olaf** accordingly looks as in Table 2.2.

As in the case of logical words, the independence of the extension of a proper name from the situation at hand results in its intension being a

[3]Strictly speaking we have only shown (or made at least plausible) that it is irrelevant for the identification of the extension of a name in a counterfactual situation who bears the name in that situation. This does not imply that it is instead the actual bearer that matters: perhaps there are quite different criteria for determining the extension of a name for a counterfactual situation. In fact, Frege had assumed this (in a notorious footnote of his essay *Über Sinn und Bedeutung*). Roughly, Frege had taken it for granted that what matters are the criteria that a speaker uses for identifying the bearer of the name. Frege's assumption is commonly thought to be problematic; we have been taking the popular counter-stance instead, as formulated by Saul Kripke in his work *Naming and Necessity* (1972).

Situation	*Extension*
s_0	Olaf
s_1	Olaf
s_2	Olaf
...	...

Table 2.2: *Tabular representation of* $[\![\textbf{Olaf}]\!]$

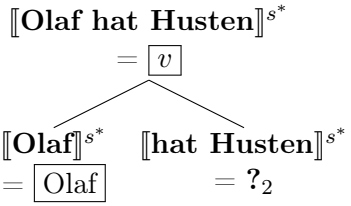
constant function, one that assigns the same value to every point in Logical Space, viz., the (actual) bearer of the name:

(6) a. $[\![\textbf{Olaf}]\!]$ = that function f with domain LS such that for any $s \in LS$ it holds that $f(s) = [\![\textbf{Olaf}]\!]^s$ = Olaf;
b. $[\![\textbf{Maria}]\!]$ = that function f with domain LS such that for any $s \in LS$ it holds that $f(s) = [\![\textbf{Maria}]\!]^s$ = Maria;
c. $[\![\textbf{Berlin}]\!]$ = that function f with domain LS such that for any $s \in LS$ it holds that $f(s) = [\![\textbf{Berlin}]\!]^s$ = Berlin;
etc.

Just like the equations (38) for interpreting the conjunctions **und** [≈ and] and **oder** [≈ or] given in the preceding chapter, (6) is to be understood as a partial specification of the lexical semantics of German. The semantic rules proper are the equations that give the intension for each name. The preceding explanation that the extension is the bearer of the name, is only supposed to help the intuitive understanding. It is not a semantic rule; for – like reference in general – who or what bears the name should flow from the intension of the name, which is being defined in this rule. Any definitional reduction of the intension to the bearer would thus be hopelessly circular. It is still common to replace the equations in (6) by a general clause to the effect that the extension of a name is its bearer, thereby sparing the seemingly redundant equations.

2.2 Subject-Predication

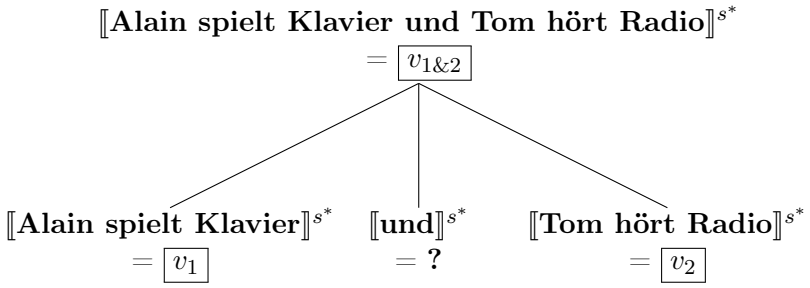
(6) certainly reduces the gap in our knowledge about the compositional interpretation of (1):

(7)


In (7) v is the truth value of (1) in situation $s^* \in LS$: $v = 1$ iff Olaf has a cough

in s^*. In order to eliminate the remaining question mark, we will assume the Principle of Compositionality and apply it backwards, as it were. According to the Principle of Compositionality, the truth value v results by combining the extension of the subject with the extension of the predicate. To determine the latter we will follow the same procedure as in the determination of the extensions of coordinating conjunctions in the preceding chapter. Using the above notation, the starting point at the time may be represented as follows:

(8)



The solution was to first capture the systematic *dependence* of the value $v_{1\&2}$ of the values v_1 and v_2. To identify this dependence, we *abstracted* from the concrete values for given sentences in given situations and looked at arbitrary alternative extensions for the clauses – which was not particularly hard, since per clause only the two truth values 0 and 1 were to be considered as alternative extensions. For each of of the four combinations of these two extensions – (v_1,v_2) = (0,0), (0,1,), (1,0), or (1,1) – a separate truth value ensues: $v_{1\&2}$ = 0, 0, 0, and 1, respectively. Since this functional dependence of the overall extension is obviously contributed by the conjunction **und** [≈ and], the obvious move was to consider it its extensions, i.e., the function that assigns to the extensions of the remaining two parts the resulting extension of the entire sentence. The same procedure also helps finding out which extension the question mark may represent in the case of (7): here, too, the extension of the entire expression – the truth value of the sentence – depends on what the extension of the other part is – i.e., who is the bearer of the name in subject position. For if this bearer happens to have a cough in the situation s^* under scrutiny, the sentence is true and thus has 1 as its extension (in s^*); otherwise its extension is 0. Again the dependence of the truth value on the bearer of the name in subject position can be represented by way of a table. What exactly this table looks like depends on the details of the situation s^* under consideration. Assuming, for concreteness, that Olaf and his sister Maria do have a cough in s^* whereas Fritz doesn't, we obtain the following functional dependence:

bearer of name	*truth value*
Maria	1
Fritz	0
Olaf	1
...	...

Table 2.3: $[\![(1)]\!]^{s^*}$ *as depending on the extension of the subject*

It can be gleaned from Table 2.3 how the extension of a sentence of the form *NN hat Husten* [≈ NN has a cough] in situation s^* depends on the extension of the subject. So if the latter is known, one can determine the former from it. The table is thus suited to represent the extension of the predicate, which is, after all, supposed to determine the sentence extension together with the subject extension – according to the Principle of Compositionality, that is. Accordingly, the extension of the predicated (in s^*) comes out as a function f that assigns a truth value $f(x)$ to each bearer x of a name – depending on whether x has a cough (in s^*):

(9) a. $[\![\textbf{hat Husten}]\!]^{s^*}(x) = \begin{cases} 1, & \text{if } x \text{ has a cough in } s^*; \\ 0 & \text{otherwise} \end{cases}$

For each bearer x of a name, (9a) declares the functional value the predicate extension assigns to x. We assume that any person, any place, any object etc. – any individual, for short – may in principle have a name and could thus play the rôle of x. The schematic equation (9a) thus defines a function whose domain consists of all individuals and whose values are always one of the two truth values, i.e., 0 or 1. The extension of the predicate **hat Husten** [≈ has a cough] thus turns out to be the characteristic function of a set M. Since this function assigns 1 to an argument x just in case x has a cough in situation s^*, the elements of M are precisely the persons that have a cough in s^*. According to our assumptions about s^* this means, in particular, that Maria $\in M$, Olaf $\in M$, but Fritz $\notin M$. Written as a list, M thus comes out as: {Maria, Olaf, ...}. Just how the three dots have to be filled in, depends on the details of s^* ab, which we do not know (because we have not specified them). There is, however, an equally common way of denoting sets with which we can characterize M precisely, viz. as: $\{x \mid x$ has a cough in $s^*\}$. Instead of the list notation used so far, this notation makes use of the so-called *set abstraction*, which also puts curly brackets around the set denotation but instead of listing of its elements has an *abstracted* variable and a *condition* (separated by a horizontal stroke). More generally:

D2.1 $\{x \mid \ldots x \ldots\}$ is that set whose elements are precisely the x for which $\ldots$ $x \ldots$ holds.

The notation defined in D2.1 is to be understood as schematic and must be

undone in any particular case where it is clear what the dots stand for. In our case '... x ...' stands for: 'x has a cough in s^*'.

In case the condition '... x ...' has the form '$x \in A$' (where A is some set), D2.1 immediately implies that the set defined by this abstraction coincides with the set A; for according to 2.1, $x \in \{x \mid x \in A\}$ holds for any x iff $x \in A$, which means that $\{x \mid x \in A\} = A$, by the Principle of Extensionality. This elementary fact, which we will sometimes exploit, has a name:

(10) *Comprehension Principle*
For all sets A it holds: $\{x \mid x \in A\} = A$.

The above determination of the predicate extension obviously does not depend on our specific example; quite generally, predicate extensions are characteristic functions. Thus, given a situation, the extension of **ist eine Insel** [≈ is an island] characterizes the set of islands in that situation; the extension of **schläft** [≈ is sleeping] is the characteristic function of the set of individuals that are asleep in the given situation, etc.:

(9) b. $[\![\textbf{ist eine Insel}]\!]^{s^*}(x) = \begin{cases} 1, & \text{if } x \text{ is an island in } s^*; \\ 0 & \text{otherwise} \end{cases}$

c. $[\![\textbf{schläft}]\!]^{s^*}(x) = \begin{cases} 1, & \text{if } x \text{ is sleeping in } s^*; \\ 0 & \text{otherwise} \end{cases}$

A notational trick helps avoiding the case distinctions in the equations (9) by directly referring to the truth value of a given statement:[4]

D2.2 If φ is a statement, then $\vdash\varphi\dashv$ is the truth value of φ; i.e.:
- $\vdash\varphi\dashv = 1$ if φ is the case; and
- $\vdash\varphi\dashv = 0$ otherwise.

Using this notation, the equations in (9) can be simplified as follows:

(11) a. $[\![\textbf{hat Husten}]\!]^{s^*}(x) = \vdash x$ has a cough in $s^*\dashv$;
b. $[\![\textbf{ist eine Insel}]\!]^{s^*}(x) = \vdash x$ is an island in $s^*\dashv$
c. $[\![\textbf{schläft}]\!]^{s^*}(x) = \vdash x$ is sleeping in $s^*\dashv$

Since sets of individuals and their characteristic functions stand in a one-one relation to each other, we will sometimes talk of the extension of a predicate *taken as a set* and use a special notation for this:

(12) a. $\downarrow[\![\textbf{hat Husten}]\!]^{s^*} = \{x \mid x \text{ has a cough in } s^*\}$;
b. $\downarrow[\![\textbf{ist eine Insel}]\!]^{s^*} = \{x \mid x \text{ is an island in } s^*\}$;
c. $\downarrow[\![\textbf{schläft}]\!]^{s^*} = \{x \mid x \text{ is sleeping in } s^*\}$.

For the general case, the notation used in (12) can be defined as follows:

[4] In other words: $\vdash\varphi\dashv$ is that truth value that is identical with 1 iff φ.

D2.3 If f is a characteristic function, then '$\downarrow f$' denotes the set characterized by f:
$\downarrow f = \{x \mid f(x) = 1\}$.

Clearly, the equations (9), (11) and (12) carry over from the random situation $s^* \in LS$ to arbitrary such situations. This also fixes the *intensions* of the predicates, which assign to arbitrary situations the extensions determined in analogy to (11). As a case in point, applying $[\![\textbf{hat Husten}]\!]$ – the intension of the predicate **hat Husten** [$\approx$ has a cough] – to a situation $s \in LS$, the functional value is a function which, again by (or in analogy to) the equation (11a), assigns to every individual x the truth value 1 iff x has a cough in s. Generalising to arbitrary situations s and individuals x we get:

(13) a. $[\![\textbf{hat Husten}]\!](s)(x) = \vdash x$ has a cough in $s\dashv$;
b. $[\![\textbf{ist eine Insel}]\!](s)(x) = \vdash x$ is an island in $s\dashv$;
c. $[\![\textbf{schläft}]\!](s)(x) = \vdash x$ is sleeping in $s\dashv$.

According to (13c), e.g., the intension of **schläft** [$\approx$ is sleeping] is a function f that assigns to each s a function f' that assigns to each x a certain truth value – namely, 1 if x is sleeping in s, and 0 otherwise: $f(s) = f'$, and $f'(x) = \vdash x$ is sleeping in $s^*\dashv$. This fixes the functional values for each argument $s \in LS$ of our three sample-predicates – and thereby their intensions.[5]

With (11a) the question mark in (7) can be eliminated, identifying the unknown predicate extension as a (characteristic) function. This also answers the question how the extension of the whole sentence is obtained from the extensions of its immediate parts: the function contributed by the predicate is applied to the argument contributed by the subject, and the resulting functional value is the extension of the sentence. We thus have another case of *functional application*, which had already been the pertinent combination in the case of sentence coordination: $[\![S]\!]^{s^*} = [\![P]\!]^{s^*}([\![NN]\!]^{s^*})$.[6] And again this equation carries over from the originally fixed situation s^* to all of Logical Space:

(14) *Compositional Determination of the Extension of Subject-Predications*
If S is a sentence with a predicate P and a proper name NN as its subject, then for all $s \in LS$ the following holds:
$[\![S]\!]^{s^*} = [\![P]\!]^{s^*}([\![NN]\!]^{s^*})$

From (13) one may derive how the proposition expressed by the sentence

[5] Quite generally any function can be fully characterized by its argument-value-pairs. This follows from the set-theoretic definition of a function and the Principle of Extensionality.

[6] Unlike predicate extensions, the relevant set-theoretic functions (intersection and union) are *binary* though, i.e., their arguments are ordered pairs (of propositions, in the case at hand). Thence the notational difference: the value of intersection for the pair consisting of the propositions p and q is traditionally written as '$p \cap q$' – instead of '$\cap\,(p,\, q)$', in analogy to other (binary) functions.

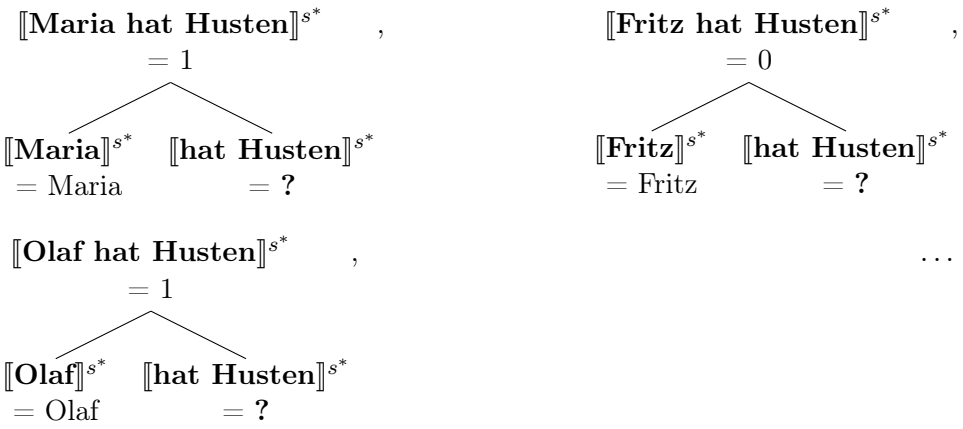
– i.e., its intension (taken as a set) – is obtained from the intensions of its parts. This will be done at the end of the chapter.

2.3 Abstraction

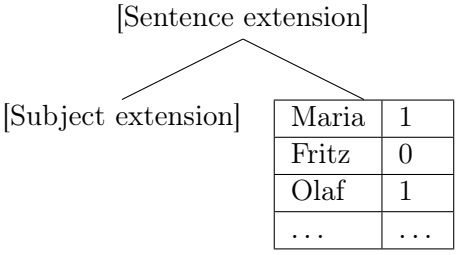
In order to determine the extension of the predicate, it did not suffice to just look at the contribution of the bearer of the name in subject position. In Table 2.3 we had instead called in *alternatives* to Olaf – bearers of other names – and determined their contribution to the resulting truth value. This procedure may be summarized in the following schema:

(15) *Identification of the Predicate Extension*

- Starting Point
  HAVE: Extension of sentence and subject; *WANT*: predicate extension:



- Construction
  Replace **?** by the function that assigns to any subject extension the corresponding sentence extension:

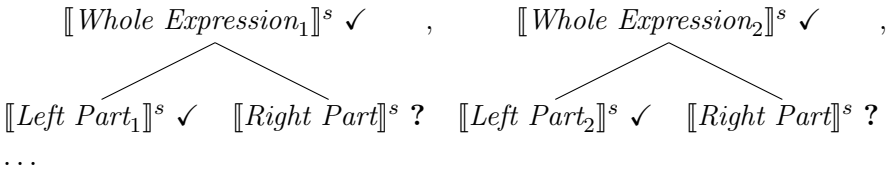


In this way one not only obtains a unified predicate extension for all predications involving the predicate **hat Husten** [≈ has a cough]. The extension of the sentence also comes out compositionally – by functional application – from the subject extensions and this common predicate extension.

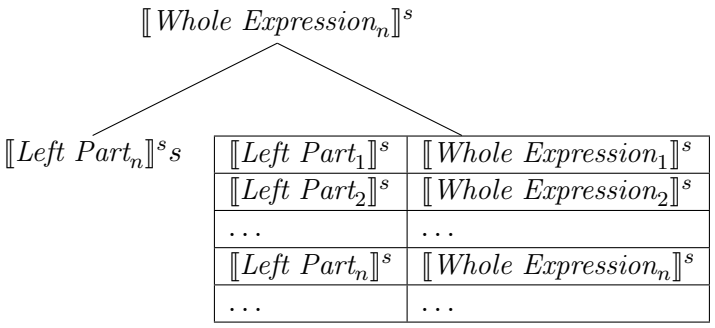
The procedure for determining the predicate extension applied in (15) is very general and cannot only be applied to predications with arbitrary predicates but also to a host of other constructions. We will always resort to it whenever we are dealing with a grammatical construction that combines two expressions, where both the extensions of the whole expression and those of one of its parts are known – i.e., the left part or the right part, as the case may be. The procedure laid out in (15) then normally allows to construct the extension of the remaining, unanalyzed part:

(16) *Identification of Unknown Extensions of Parts of Expressions*

- Starting Point
  HAVE (✓): Extensions of the whole expressions and their left parts;
  WANT (**?**): Extension of the right part:



- Construction
  Replace **?** by the function that assigns to the known extensions of the left part the corresponding extensions of the whole expression:



The procedure can, of course, also be applied if it happens to be the left part whose extension is unknown; in the next section we will encounter such a case. And it can also be applied if more than two parts are involved in the grammatical construction – as long as there is only one part whose extension is still unknown.[7] Thus in the preceding chapter we had analyzed sentence coordination as a ternary construction, and presupposed that, like the whole expression, two of its three parts had truth values as extensions.

[7] ... in the sense that it is not known what *kind* of object makes the compositional contribution to the extension of the whole expression. As native speakers, the meanings are, of course, known and familiar to us in that we have implicit mastery over them.

In retrospect the analysis of the conjunctions **und** [≈ and] and **oder** [≈ or] thus turns out to be a variant of the abstraction procedure in (15).

In formal logic the construction given in (15) and (16) is called *functional abstraction*, because the unknown extension so constructed *abstracs* from the concrete contributions of the individual parts already analyzed – the proper names in (15) – and reduces them to a common pattern, the *function* so constructed. In a sense, functional abstraction reverses functional application: if application is a kind of addition of extensions, then abstraction amounts to subtraction. This kind of subtraction of functional abstraction is one of the most important analytic tools of logical semantics. It is primarily its flexibility that motivates the top-down direction of analysis pursued here, starting with the sentence, going down to its immediate parts and their immediate parts, and all the way down to the words.[8]

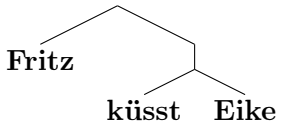
2.4 Object-Predication

It is an advantage of the strategy described in (16) that it can also be employed if the extensions known at Starting Point have themselves been obtained by functional abstraction. Once it is known that predicate extensions are certain (characteristic) functions, this knowledge can be exploited to derive the extensions of complex predicates. Here is a case in point:

(17) **Fritz küsst Eike.**
[≈ Fritz is kissing Eike]

We take it that **Fritz** is the subject and hence (17) is a predication. Decomposing (17) into its parts and parts of parts is again straightforward:

(18)



We already know the extensions of the whole sentence and its immediate parts:

(19) a. $[\![(17)]\!]^{s^*} = \vdash$Fritz is kissing Eike in $s^*\dashv$
b. $[\![\textbf{Fritz}]\!]^{s^*} =$Fritz

[8]The strategy ultimately goes back to Gottlob Frege's essay *Funktion und Begriff* [≈ Function and Concept] (1891) and extends the *Context Principle* from Frege's *Grundlagen der Arithmetik* [≈ Foundations of Arithmetic] (1884), according to which the meaning of an expression is construed as its contribution to sentence meaning.

c. $[\![\textbf{küsst Eike}]\!]^{s^*}$ =

Fritz	⊢Fritz is kissing Eike in s^*⊣
Maria	⊢Maria is kissing Eike in s^*⊣
Eike	⊢Eike is kissing herself in s^*⊣
...	...
x	⊢x is kissing Eike in s^*⊣

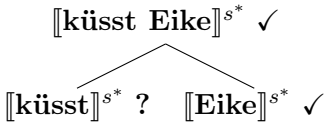
.

Since the predicate is a complex expression, its extension ought emerge compositionally from the extensions of its immediate parts. One of these extensions is already known to us: that of the proper name **Eike**; for we assume that this extension is independent of whether the name is in subject or object position:

(20) $[\![\textbf{Eike}]\!]^{s^*}$ =Eike

However, we do not yet know the extension of the transitive verb **küsst** [≈ is kissing]. But since we do know both the extension (19c) of the whole predicate and the extension (20) of the object, we may now perform functional abstraction:

(21)



For a start, we need to consider alternative expressions again, where the part that has already been analyzed – the object **Eike** – is replaced by other expressions of the same kind that have also been analyzed – other proper names, that is. As in (19c), we then obtain functions that may be represented by way of tables:

(22) $[\![\textbf{küsst Fritz}]\!]^{s^*}$ =

Fritz	⊢Fritz is kissing himself in s^*⊣
Maria	⊢Maria is kissing Fritz in s^*⊣
Eike	⊢Eike is kissing Fritz in s^*⊣
...	...
y	⊢y is kissing Fritz in s^*⊣

$[\![\textbf{küsst Maria}]\!]^{s^*}$ =

Fritz	⊢Fritz is kissing Maria in s^*⊣
Maria	⊢Maria is kissing herself in s^*⊣
Eike	⊢Eike is kissing Maria in s^*⊣
...	...
y	⊢y is kissing Maria in s^*⊣

$[\![\textbf{küsst Eike}]\!]^{s^*}$ =

Fritz	⊢Fritz is kissing Eike in s^*⊣
Maria	⊢Maria is kissing Eike in s^*⊣
Eike	⊢Eike is kissing herself in s^*⊣
...	...
y	⊢y is kissing Eike in s^*⊣

...

$[\![\textbf{küsst}\,NN]\!]^{s^*} =$

Fritz	$\vdash$Fritz is kissing NN in $s^*\dashv$
Maria	$\vdash$Maria is kissing NN in $s^*\dashv$
Eike	$\vdash$Eike is kissing NN in $s^*\dashv$
...	...
y	$\vdash y$ is kissing NN in $s^*\dashv$

By the general method (16), one may now construct a function that assigns to the extensions of the objects – the bearers of the names – the corresponding predicate extensions in (22); and this function is then the extension of the transitive verb **küsst** [≈ is kissing]:

(23) $[\![\textbf{küsst}]\!]^{s^*} =$

Fritz	Fritz	$\vdash$Fritz is kissing himself in $s^*\dashv$
	Maria	$\vdash$Maria is kissing Fritz in $s^*\dashv$
	Eike	$\vdash$Eike is kissing Fritz in $s^*\dashv$
	...	...
	y	$\vdash y$ is kissing Fritz in $s^*\dashv$
Maria	Fritz	$\vdash$Fritz is kissing Maria in $s^*\dashv$
	Maria	$\vdash$Maria is kissing herself in $s^*\dashv$
	Eike	$\vdash$Eike is kissing Maria in $s^*\dashv$
	...	...
	y	$\vdash y$ is kissing Maria in $s^*\dashv$
Eike	Fritz	$\vdash$Fritz is kissing Eike in $s^*\dashv$
	Maria	$\vdash$Maria is kissing Eike in $s^*\dashv$
	Eike	$\vdash$Eike is kissing herself in $s^*\dashv$
	...	...
	y	$\vdash y$ is kissing Eike in $s^*\dashv$
...		
x	Fritz	$\vdash$Fritz is kissing x in $s^*\dashv$
	Maria	$\vdash$Maria is kissing x in $s^*\dashv$
	Eike	$\vdash$Eike is kissing x in $s^*\dashv$
	...	...
	y	$\vdash y$ is kissing x in $s^*\dashv$
...		
y	Fritz	$\vdash$Fritz is kissing y in $s^*\dashv$
	Maria	$\vdash$Maria is kissing y in $s^*\dashv$
	Eike	$\vdash$Eike is kissing y in $s^*\dashv$
	...	...
	y	$\vdash y$ is self-kissing in $s^*\dashv$
...		

The table may appear confusing, but it is constructed by a simple principle. In the leftmost column one finds the possible object extensions that are each assigned corresponding the predicate extension to their right. And again, the semantic combination that combines the function represented in (23) with

the extension of the object is is functional application:

(24) *Compositional Determination of the Extensions of Direct-Object-Predications*
If P is a predicate consisting of a transitive verb V and a proper name NN as its direct object, then for all $s \in LS$ the following holds: $[\![P]\!]^s = [\![V]\!]^s([\![NN]\!]^s)$.

Using the extension of **küsst** [≈ is kissing] in (23) and the combination (24), the extension of a sentence like **Maria küsst Fritz** [≈ Maria is kissing Fritz] can now be derived compositionally on the basis of the interpretation of subject-predication as developed in Section 2.2:

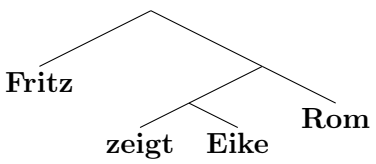
(25) $[\![\textbf{Maria küsst Fritz}]\!]^{s^*}$
$\overset{(14)}{=}$ $[\![\textbf{küsst Fritz}]\!]^{s^*}([\![\textbf{Maria}]\!]^{s^*})$
$\overset{(5)}{=}$ $[\![\textbf{küsst Fritz}]\!]^{s^*}(\text{Maria})$
$\overset{(24)}{=}$ $[\![\textbf{küsst}]\!]^{s^*}([\![\textbf{Fritz}]\!]^{s^*})(\text{Maria})$
$\overset{(5)}{=}$ $[\![\textbf{küsst}]\!]^{s^*}(\text{Fritz})(\text{Maria})$
$\overset{(23)}{=}$ ⊢Maria is kissing Fritz in s^*⊣

The last line of these equations holds because the function represented in (23) assigns to the bearer of the name **Fritz** – i.e., Fritz – a function which applied to Maria (the bearer of the pertinent name) yields said truth value (in the second line). It should be noted that the derivation in (25) only makes use of the equations (5) [for the proper names **Fritz** and **Maria**] and (23) [for the verb **küsst**] as well as the compositional rules (14) [for the subject-predication] and (24) [for the object-predication]. This explains how the truth value of a the sentence emerges from the extensions of its lexical parts and the grammatical construction solely on the basis of general principles.

The strategy employed for identifying the extensions of predicates and transitive verbs can also be applied to verbs that take more than one object. Thus one may derive the extension of the ditransitive verb **zeigen** [≈ show] by functional abstraction, using examples like (26) and assuming the structure (27).

(26) **Fritz zeigt Eike Rom.**
[≈ Fritz is showing Rome to Eike.]

(27)



Since the complex verb **zeigt Eike** [≈ is showing Eike] behaves like a transitive verb in that it may form a predicate with an (accusative) object, we may assume that, like the extension of **küsst** [≈ is kissing] represented in (23), its extension is a function that assigns predicate extensions to arbitrary individuals. Accordingly the method (16) for determining extensions can be applied to this expression; for apart from the extension of the whole expression, the extension of the object **Eike** is known, too. As a result one obtains a function whose tabular representation should really be forbidden for reasons of space and transparency; but the section relevant for the example looks like this:

(28) $[\![\textbf{zeigt}]\!]^{s^*} =$

...	...		
Eike	...	...	
	Rome	...	...
		Fritz	⊢ in s^* Fritz is showing Rome to Eike⊣
		...	...
	...	...	
...	...		

2.5 Lambda-Terms

The idea behind the analyses (21) and (27) of transitive and ditransitive verbs generalizes to verbs with arbitrarily many (nominal) objects. As the number of objects increases though, the tabular representation of the extension of the verb gets more involved and confusing. But despite this monstrous expansion, the structure of the tables remains straightforward; we had already remarked this in connection with (23). (28) is no different in this respect, as one may verify by going through a single 'line' from left to right, finally arriving at the truth value in the innermost box: the outermost column corresponds to the indirect object (MM), and once the indirect object has been fixed, the second column to the left of the outermost box matches the direct object (LL), etc.; at the end the truth value of a sentence of the form '*NN* **zeigt** *MM LL*' [≈ NN shows LL to MM] in s^* emerges.

The simple structure of tables like (23) and (28) can be exploited to arrive at a much more compact representation of functions than the table format. This representation may first be applied to predicate extensions and then adapted to more complex cases. Above we had given an analysis of the predicate **hat Husten** [≈ has a cough] that we repeat here in a slightly more detailed form:

(29) $[\![\textbf{hat Husten}]\!]^{s^*} =$

Maria	1 [= ⊢Maria has a cough in s^*⊣]
Fritz	0 [= ⊢Fritz has a cough in s^*⊣]
Olaf	1 [= ⊢Olaf has a cough in s^*⊣]
...	...
x	⊢x has a cough in s^*⊣
...	...

The individual called x in the highlighted line is the bearer of an arbitrary name. Due to this arbitrariness, the marked line covers all other lines: the first three lines are special cases of the marked line; for the variable 'x' stands in for arbitrary bearers of names, thus including Fritz, Maria, and Olaf. And what is right for the first three lines of the table in (29), also holds for all other lines: they are already accounted for by the marked line. It thus suffices to write down – instead of the whole table – the marked line only, which in turn has two parts: the variable that stands for arbitrary arguments of the function and the description the truth value that depends on the individual denoted by the variable. In semantics a peculiar notation has become common for this, where a small Greek letter Lambda (like *l*eft) indicates the argument and is separated from the functional value by a dot:[9]

(30) $[\![\textbf{hat Husten}]\!]^{s^*} = \lambda x.\vdash x$ has a cough in $s^*\dashv$

We will from now on use such *lambda-terms* as in (30) to describe functions. A brief comparison of (29) and (30) already shows that not only space is saved, but transparency is gained too, by using this representation; for the lambda-term summarizes exactly what the various lines in (29) have in common and thus reflects the construction principle underlying the table.

More complex functions, too, may be described by lambda-terms. Since each of the functional values of the extension of **küsst** [≈ is kissing] defined in (23) is itself a function, the table may first be simplified like this:

(31) $[\![\textbf{küsst}]\!]^{s^*} =$

Eike	$\lambda x.\vdash x$ is kissing Eike in $s^*\dashv$
Fritz	$\lambda x.\vdash x$ is kissing Fritz in $s^*\dashv$
Maria	$\lambda x.\vdash x$ is kissing Maria in $s^*\dashv$
...	...

The result is again a function, whose typical line is of the following form:[10]

[9] The representation of functions by terms can already be found in Frege's 'Über Funktion und Begriff' [≈ On Function and Concept] (1891), where a different notation is used. Lambda-terms have been introduced by the US logicians Alonzo Church and Stephen Kleene in the 1930s.

[10] It should be noted that we need to use a different variable ('y') to denote the extension of the object, so as to avoid a conflict with the variable for the subject extension ('x').

(32)

...	...
y	$\lambda x.\vdash x$ is kissing y in $s^*\dashv$
...	...

Hence the whole extension of **küsst** [≈ is kissing] can be accounted for by *a single* lambda-term:

(33) $[\![\textbf{küsst}]\!]^{s^*} = \lambda y.\lambda x.\vdash x$ is kissing y in $s^*\dashv$

In (33) the two *lambda-prefixes* now take the place of the embedded tables in (23). In a similar way, even more deeply embedded tables like the one indicated in (28) can be reduced to a single line consisting of a lambda-term – as will be shown in an exercise.

Lambda-terms are (meta-linguistic) expressions denoting functions whose use is fixed by a notational convention:

D2.4 An expression of the form '$\lambda x. \ldots\ x\ \ldots$' stands for a function f that assigns to any arguments x the value $f(x) = \ldots\ x\ \ldots$.

D2.4 is *schematic* in that any concrete instance of the convention depends on what the part following the prefix '$\lambda x.$' is – the so-called *matrix* of the lambda-term. The dots are meant to indicate that it is some description that makes use of the variable 'x' *bound* in the prefix. This variable may, of course, occur arbitrarily many times – once, twice, ... – or even not at all. (This last case will be addressed in an exercise.) According to D2.4, then, both '$\lambda x.2x$' and '$\lambda x.x + x$' stand for the function of doubling a number.

D2.4 leaves open what the domain of a function denoted by a lambda-term is: what does 'x' stand for? In actual practice this will usually be clear from the context in which the lambda-term is used. In Chapter 5 we will introduce a somewhat more precise notation that will be unambiguous in this respect.

Describing functions by lambda-terms not only saves place and creates clarity vis-à-vis tabular representations, it also simplifies the compositional account of extensions of complex terms by functional application. This is best seen by way of an example. In the last step of the above derivation (25) of the extensions of **Maria küsst Fritz** [≈ Maria is kissing Fritz] (in s^*) we applied the function $[\![\textbf{küsst}]\!]^{s^*}$ to Fritz and then the result to Maria:

(25)
... $[\![\textbf{küsst}]\!]^{s^*}$(Fritz)(Maria)
$\overset{(23)}{=}$ $\vdash$Maria is kissing Fritz in $s^*\dashv$

This transition simply relies on the table (23) that gives the extension of **küsst** [≈ kissing[(in s^*), which can be applied to Fritz to give a function that yields the resulting truth value when applied to Maria. The compact representation by lambda-terms now allows for the function to be denoted

directly. Plugging in the lambda-term from (33) (and bracketing it, for readability), the penultimate line of (25) comes out like this:

(25′) $[\lambda y.\lambda x.\vdash x \text{ is kissing } y \text{ in } s^*\dashv](\text{Fritz})(\text{Maria})$

The function denoted by the lambda-term is first applied to Fritz. The intermediate result thus obtained is again a function that can be represented by a lambda-term:

(25″) $[\lambda y.\lambda x.\vdash x \text{ is kissing } y \text{ in } s^*\dashv](\text{Fritz})(\text{Maria})$
$=$ $[\lambda x.\vdash x \text{ is kissing Fritz in } s^*\dashv](\text{Maria})$

Finally this function is applied to the subject-extension – with the familiar result. All in all, the derivation (25) looks like this when done with lambda-terms:

(25‴) $[\![\textbf{Maria küsst Fritz}]\!]^{s^*}$
$\overset{(14)}{=}$ $[\![\textbf{küsst Fritz}]\!]^{s^*}([\![\textbf{Maria}]\!]^{s^*})$
$\overset{(5)}{=}$ $[\![\textbf{küsst Fritz}]\!]^{s^*}(\text{Maria})$
$\overset{(24)}{=}$ $[\![\textbf{küsst}]\!]^{s^*}([\![\textbf{Fritz}]\!]^{s^*})(\text{Maria})$
$\overset{(5)}{=}$ $[\![\textbf{küsst}]\!]^{s^*}(\text{Fritz})(\text{Maria})$
$=$ $[\lambda y.\lambda x.\vdash x \text{ is kissing } y \text{ in } s^*\dashv](\text{Fritz})(\text{Maria})$
$=$ $[\lambda x.\vdash x \text{ is kissing Fritz in } s^*\dashv](\text{Maria})$
$=$ $\vdash \text{Maria is kissing Fritz in } s^*\dashv$

The striking feature of the last two transitions is that each time one lambda-prefix was eliminated while the corresponding variable in the remainder of the lambda-term was replaced by the argument. First 'λy' disappears while 'Fritz' takes the place of 'y'; then 'λx' disappears and 'Maria' replaces 'x'.[11] This is not accidental. In the lambda-term the variable mentioned in the prefix precisely stands in for an arbitrary argument. So if the function denoted by a lambda-term is applied to a specific argument, the result may be described by having the designation of the argument occupy the place of the variable; and of course, the lambda-prefix vanishes, since the result is a functional *value* – i.e., that which is described to the right of the dot. This elimination of lambda-prefixes with a subsequent insertion of arguments is usually called *λ-conversion.*[12] Schematically (and somewhat sloppily) it may be captured as follows:

[11] We put the names 'Fritz' and 'Maria' in quotes because it is not the persons that take the place of variables after all; the discussion, then, is meta-metalinguistic!

[12] This is the term most common in semantics. In logic (and computer science) one rather speaks of *β-conversion* or – if only one direction of the reformulation is meant – *β-reduction.*

(34) *λ-Conversion*
$[\lambda x. \ldots\ x\ \ldots](a) = \ldots\ a\ \ldots$

We will give a more precise formulation of the principle in Chapter 5. λ-conversion plays a crucial rôle in semantic practice. For it makes it possible to compute compositional derivations in a quasi-mechanical way. This largely simplifies testing semantic analyses by concrete examples: by erasing lambda-prefixes and inserting arguments one may determine which extension a given analysis predicts for a given complex expression (in a given situation) and then compare it to the native understanding of the expression. The construction and evaluation of lambda-terms make a large part of semanticists' everyday work. Thence the (serious) autobiographic confession of the American semanticist Barbara Partee: *Lambdas changed my life.*

There is a certain relation between the notational conventions introduced in D2.1, D2.3, and D2.4, which shows in lambda-terms for characteristic functions like the one in (30):

(35) $\lambda x. \vdash x$ has a cough in $s^* \dashv$

The term in (35) describes the extension of the predicate **hat Husten** [≈ has a cough] (in s^*). As we saw in (11a), this extension is the characteristic function of the set (of individuals) given in (36):

(36) $\{x \mid x$ has a cough in $s^*\}$

The lambda-term in (35) thus denotes the characteristic function of the set defined by set abstraction in (36); and the condition used there happens to be the matrix of the lambda term. This connection is perfectly general: whenever a lambda-term denotes a characteristic function, the set characterized by the latter can be defined by set abstraction, using the same variable and the matrix as the condition:

(37) $\downarrow [\lambda x. \vdash \ldots\ x\ \ldots \dashv] = \{x \mid \ldots\ x\ \ldots\}$

The connection (37) makes between the notations defined in D2.1, D2.3, and D2.4 is readily seen. For according to D2.1, any object a is a member of the set $\{x \mid \ldots\ x\ \ldots\}$ if, and only if, $\ldots\ a\ \ldots$ holds. But then according to D2.3, a is an element of $\downarrow[\lambda x. \vdash \ldots\ x\ \ldots \dashv]$ just in case $[\lambda x. \vdash \ldots\ x\ \ldots \dashv](a) = 1$ – which, by λ-conversion means precisely: $\vdash \ldots\ a\ \ldots \dashv = 1$, i.e., that $\ldots\ a\ \ldots$ holds. So for any a : $a \in \{x \mid \ldots\ x\ \ldots\}$ iff $a \in \downarrow [\lambda x. \vdash \ldots\ x\ \ldots \dashv]$ – from which (37) follows, in view of the Principle of Extensionality.

Disregarding the difference between sets and their characteristic functions, (37) shows that set abstractions are a kind of notational variant of lambda-terms. It has to be noticed, though, that the equation (37) only holds for lambda-terms that denote characteristic functions in the first place; for other terms, the notion of the set characterized, indicated by the down-

arrow defined in D2.2, does not make any sense in the first place. Thus any set abstraction can be 'mimicked' by a corresponding lambda-term, but the reverse does not hold; for not every lambda-term denotes a characteristic function. As a case in point, the values assigned by the extensions of transitive and ditransitive verbs represented by lambda-terms in (38) are of course functions themselves – and not truth values:

(38) a. $[\![\textbf{küsst}]\!] = \lambda y.\lambda x.\vdash x \text{ is kissing } y \text{ in } s^*\dashv$
b. $[\![\textbf{zeigt}]\!] = \lambda z.\lambda y.\lambda x.\vdash x \text{ is showing } y \text{ to } z \text{ in } s^*\dashv$

2.6 Compositionality of Intensions

At the end of Section 2.2 we claimed that the intensions introduced before behave compositionally; lambda-terms are of great help in establishing this observation, which is what we will do now. Since the intension is the extension in its functional dependence on the situation, it may also be represented by a lambda-term. We thus have, for any expression A:

(39) a. $[\![A]\!] = \lambda s.[\![A]\!]^s$

(39a) implies an elementary fact, which is nevertheless crucial for what follows:[13]

(39) b. $[\![A]\!](s) = [\![A]\!]^s$, for all situations $s \in LS$.

Let us first recall what the various intensions defined in this chapter look like by listing some of the expressions analyzed above in this notation:

(40) a. $[\![\textbf{Olaf}]\!] = \lambda s.\text{Olaf}$
b. $[\![\textbf{hat Husten}]\!] = \lambda s.\lambda x.\vdash x \text{ has a cough in } s\dashv$
c. $[\![\textbf{küsst}]\!] = \lambda s.\lambda y.\lambda x.\vdash x \text{ is kissing } y \text{ in } s\dashv$
d. $[\![\textbf{zeigt}]\!] = \lambda s.\lambda z.\lambda y.\lambda x.\vdash x \text{ is showing } y \text{ to } z \text{ in } s\dashv$

As has become clear in the first section, in the case of proper names we are dealing with constant functions. The lambda-term (40a) brings this out clearly: the bound variable 's' does not even occur in its matrix, which is why the value cannot depend on the argument it denotes. The extension of predicates, on the other hand, does normally vary across Logical Space: who has a cough depends on the circumstances, the situation, and thus the extensions of the predicate **hat Husten** [≈ has a cough] frequently (though not always) characterizes different sets of individuals in different situations.

[13] In fact, (39a) and (39b) are logically equivalent: due to λ-conversion, any situation $s^* \in LS$ satisfies: $[\lambda s.[\![A]\!]^s](s^*) = [\![A]\!]^{s^*}$. Thus (39b) expresses that $[\![A]\!]$ coincides with $[\lambda s.[\![A]\!]^s]$ on all arguments $s \in LS$ and is thus the same function (due to the Principle of Extensionality) – as claimed in (39a).

The same holds for other types of verbs: who is kissing whom; who is showing what to whom; etc. all depends on the situation at hand.

A simple example should now make clear how the *intensions* given in (40) determine the intensions of the complex expressions in which they occur, viz., solely due to the above compositional combinations of the corresponding *extensions*. The following sentence is a case in point:

(41) **Olaf küsst Maria.**
[≈ Olaf is kissing Maria]

The intension of (41) is a function that assigns to every situation $s \in LS$ the extension of (41) in s. The latter may in turn be determined by the general compositional rule (14) from Section 2.2:

(14) *Compositional Determination of the Extension of Subject-Predications*
If S is a sentence with a predicate P and a proper name NN as its subject, the for all $s \in LS$ the following holds:
$[\![S]\!]^s = [\![P]\!]^s([\![NN]\!]^s)$

From (14) and (39a) we conclude:

(42) $[\![\textbf{Olaf küsst Maria}]\!]$
$= \lambda s.[\![\textbf{Olaf küsst Maria}]\!]^s$
$= \lambda s.[\![\textbf{küsst Maria}]\!]^s([\![\textbf{Olaf}]\!]^s)$

The extensions of the predicate (for arbitrary situations s) mentioned in the last line may in turn be determined from the extensions of the verb and the object, following the compositional rule (24) from Section 2.4:

(24) *Compositional Determination of the Extensions of Direct-Object-Predications*
If P is a predicate consisting of a transitive verb V and a proper name NN as its direct object, then for all $s \in LS$ the following holds:
$[\![P]\!]^s = [\![V]\!]^s([\![NN]\!]^s)$.

(24) can now be used to extend the chain of equations in (42):

(42′) $[\![\textbf{Olaf küsst Maria}]\!]$
$= \ldots$
$= \lambda s.[\![\textbf{küsst}]\!]^s([\![\textbf{Maria}]\!]^s)([\![\textbf{Olaf}]\!]^s)$
$= \lambda s.[\![\textbf{küsst}]\!](s)([\![\textbf{Maria}]\!](s))([\![\textbf{Olaf}]\!](s))$

The final transition is due to (39b). Now the equations (40) apply; for clarity we rename the variables (whose names are arbitrary anyway) and add a few brackets:

(42″) $[\![\textbf{Olaf küsst Maria}]\!]$

$=$ ...
$=$ $\lambda s.[\lambda t.\lambda y.\lambda x.\vdash x$ is kissing y in $s\dashv](s)([\lambda u.\text{Maria}](s))([\lambda v.\text{Olaf}](s))$

The confusingly many lambdas in (42″) may now be largely eliminated by a series of lambda-conversions; in converting the term prefixed with 'λt', we replace the argument by 's', whereas only the (constant) matrix remains of the other two terms, prefixed with 'Maria' bzw. 'Olaf'; for variables that do not occur need not be replaced:

(42*)
$=$ ...
$=$ $\lambda s.[\lambda y.\lambda x.\vdash x$ is kissing y in $s\dashv](\text{Maria})(\text{Olaf})$

Now the lambda-term beginning with 'λy' with the argument 'Maria' can be converted:

(43) $[\![$**Olaf küsst Maria**$]\!]$
$=$ ...
$=$ $\lambda s.[\lambda x.\vdash x$ is kissing Maria in $s\dashv](\text{Olaf})$
$=$ $\lambda s.\vdash$Olaf is kissing Maria in $s\dashv]$

The final step in (43) converts the 'λx'-term. The result is a description of the intension of the whole sentence that – correctly – turns out to be the characteristic function of the set of situations in which Olaf is kissing Maria.

The derivation of (43) is solely based on the equations in (40) and the pertinent compositional rules (14) and (24) for the extensions. Under the (simplifying) assumption that the expressions analyzed in (40) are lexical, one may regard the information captured in the equations as part of the (German) lexicon and thus implicitly known to competent speakers.[14] The example illustrates that, for a systematic derivation of the intension of the entire sentence, it suffices to know how extensions combine compositionally. The speakers, who ought to know the intensions – but not necessarily the extensions – of all expressions, can determine them solely from the (largely unsystematic) lexical intensions and the rules of extensional combination. This connection does not only hold for the predication constructions considered above but can be shown for all constructions in which the extensions can be determined compositionally. In the following we will not point this out explicitly and content ourselves with extensional principles of compositionality, trusting that this implies the compositionality of intensions, which is crucial to an explanation of systematic linguistic knowledge.

[14] Of the four equations in (40), only (40a) is clearly lexical, though only part of the idiolects of those speakers that use the proper name **Olaf** in the reading here assumed. (38b–d) are not lexical in view of the inflectional morphology, which we are neglecting here.

2.7 Exercises for Chapter 2

A1 Show that the analysis of the conjunctions **und** [≈ and] and **oder** [≈ or] given in the previous chapter is a variant of the abstraction procedure (16) for finding unknown extensions.

A2 Find the extensions of the parts and parts of parts of the following sentences by applying the abstraction method. Assume only the extensions of the occurring names and the entire sentences as known. Construct the extensions first with the help of tables, and then derive the corresponding lambda-terms.

I **Kevin schläft.**
[≈ Kevin is sleeping.]

II **Marcel sieht Kevin.**
[≈ Marcel is seeing Kevin.]

III **Marcel haut Jacqueline.**
[≈ Marcel is beating Jacqueline.]

IV **Jacqueline zeigt Marcel Kevin.** [Jacqueline is showing Kevin to Marcel.]

A3 Define the intensions of *every* expression in A2.

A4 Reduce the list notation for sets to set abstraction by showing that, for any definition of a set M by listing its members there is a representation of M in terms of set abstraction (but without listing).

A5 Determine the extension of **Michel überstellt Alfred Ida** [≈ Michel is handing over Ida to Alfred] for an arbitrary situation s^* in the style of (25‴). In doing so, formulate a rule (along the lines of (24)) for compositionally determining the extensions of predicates with direct and indirect objects.

A6 Show that the intension of **Michel überstellt Alfred Ida** [≈ Michel is handing over Ida to Alfred] results compositionally form the intensions of its parts.

Chapter 3

Quantification

3.1 Quantifying Noun Phrases

In the preceding chapter we had only considered sentences whose subject positions were occupied by proper names. On the background of our findings there, we now turn to the interpretation of other subjects.

(1) **Niemand hustet.**
[≈ Nobody is coughing.]

It is clear that **niemand** [≈ nobody] is not a proper name, and it certainly does not refer to a single individual. So the above semantics does not apply to this kind of noun phrase. We will instead try to construct its extension using the abstraction method. First of all we convince ourselves that all preconditions are fulfilled: on the one hand, we know the extension of the whole sentence – its truth value; on the other hand, we also know the extension of the other constituent, for we just identified the extensions of predicates: the extension of **hustet** [≈ is coughing] came out as a function that assigns truth values to individuals – thus the characteristic function of a set of individuals. According to the abstraction procedure this should suffice to identify the extension of **niemand** [≈ nobody]. We just need to consider arbitrary variants of (1) that only differ in their predicates, and then pair off the extensions of the latter with the corresponding resulting truth values:

(1′) **Niemand schläft.**
[≈ Nobody is sleeping.]

(1″) **Niemand ist eine Insel.**
[≈ Nobody is an island.]

The predicate extensions can again be given by lambda-terms:

(2) $[\![\textbf{hustet}]\!]^{s^*} = \lambda x.\vdash x \text{ is coughing in } s^* \dashv$

(2′) $[\![\textbf{schläft}]\!]^{s^*} = \lambda x.\vdash x \text{ is sleeping in } s^*\dashv$

(2″) $[\![\textbf{ist eine Insel}]\!]^{s^*} = \lambda x.\vdash x \text{ is an island in } s^*\dashv$

The equations (2)–(2″) hold for any possible situation $s^* \in LS$. To study the interaction of the various extensions in detail, let us consider a specific situation s^* – a day with the Müller family at the beach of Palma: everybody is in good health, the parents Horst and Gaby are asleep, while the kids Max and Susi are busy with the construction of a sand castle. Let us suppose that the function defined in (2) yields the value 0 for any individual in s^* – the members of the Müller family, the other beachgoers, their basket chairs, the grains of sand, etc. – because everyone is so healthy. The value of the extension of **schläft** [≈ is sleeping] defined in (2′), though, equals 1 for at least two arguments: Horst and Gaby Müller; for simplicity we assume that the two are the only sleepers in s^*. Finally, Mallorca is the only island around – and thus the only object for which the value of the extension in (2″) is the truth value 1. Given these assumptions about s^*, the sets characterized by (2)–(2″) are:

(3) $\downarrow [\![\textbf{hustet}]\!]^{s^*} = \emptyset$

(3′) $\downarrow [\![\textbf{schläft}]\!]^{s^*} = \{\text{Gaby}, \text{Horst}\}$

(3″) $\downarrow [\![\textbf{ist eine Insel}]\!]^{s^*} = \{\text{Mallorca}\}$

So much for the predicate extensions. The extensions of the sentences (1)–(1″), their truth values in the same situation s^*, are obvious:

(4) $[\![(1)]\!]^{s^*} = [\![\textbf{Niemand hustet}]\!]^{s^*} = 1;$

(4′) $[\![(1')]\!]^{s^*} = [\![\textbf{Niemand schläft}]\!]^{s^*} = 0;$

(4″) $[\![(1'')]\!]^{s^*} = [\![\textbf{Niemand ist eine Insel}]\!]^{s^*} = 1.$

By pairing off predicate extensions with the corresponding sentence extensions, we obtain the following table, which according to the abstraction procedure, should represent the extension of the subject:

(5) $[\![\textbf{niemand}]\!]^{s^*} =$

$\lambda x.\vdash x$ is coughing in $s^*\dashv[= [\![\textbf{hustet}]\!]^{s^*}]$	$= 1[= [\![(1)]\!]^{s^*}]$
$\lambda x.\vdash x$ is sleeping in $s^*\dashv[= [\![\textbf{schläft}]\!]^{s^*}]$	$= 0[= [\![(1')]\!]^{s^*}]$
$\lambda x.\vdash x$ is an island in $s^*\dashv[= [\![\textbf{ist eine Insel}]\!]^{s^*}]$	$= 1[= [\![(1'')]\!]^{s^*}]$
...	...

An inspection of each of the sets (3)–(3″) characterized by the corresponding predicate extensions makes the construction principle underlying table (5) apparent: a sentence with **niemand** [≈ nobody] as its subject is true just

in case this set does not contain any person.[1] Thus if X is a predicate extension, $[\![\textbf{niemand}]\!]^{s^*}$ assigns to X the truth value 1 just in case the set characterized by X does not overlap with the set Per_{s^*} of persons in s^*:

(6) $[\![\textbf{niemand}]\!]^{s^*}(X) = 1$ iff $\downarrow X \cap Per_{s^*} = \emptyset$

Given the lambda-notation, the equation (6), which holds for any predicate extension X and situation s^*, can be reformulated thusly:[2]

(7) $[\![\textbf{niemand}]\!]^{s^*} = \lambda X.\vdash\downarrow X \cap Per_{s^*} = \emptyset\dashv$

Since the extension of **niemand** [≈ nobody] has been constructed by abstraction, the compositionality of the extensions is once more guaranteed: the truth value of a sentence S formed by putting a quantifying noun phrase in subject position derives from applying the extension of the subject QN to the extension of the predicate P, i.e., by functional application:

(8) *Compositional Determination of the Extension of Subject-Quantifications*
If S is a sentence with a predicate P and a quantifying noun phrase QN as its subject, then the following holds for all $s \in LS$:
$[\![S]\!]^s = [\![QN]\!]^s([\![P]\!]^s)$.

It should be noted that the direction of functional application has been reversed vis-à-vis the subject-predications analyzed in the preceding chapter (under (37)); for there it was the predicate extension that was applied to the extension of the subject, wheres in quantifications it is the other way round. This difference is grammatically conditioned in the sense that we assume (like many syntacticians) that predication and quantification are distinct constructions. At the end of the chapter we will, however, present an alternative analysis that does without assuming such a difference.

Given (8), the intension **niemand** [≈ nobody] is now fixed too; for obviously (7) applies independently of the specific situation s^*:

(9) $[\![\textbf{niemand}]\!] = \lambda s.\lambda X.\vdash\downarrow X \cap Per_s = \emptyset\dashv$

There are quite a few noun phrases whose semantic extensions and intension can be constructed in a similar way as that of **niemand** [≈ nobody]. We only look at one example and leave further cases to an exercise and later sections:

[1] Maybe pets should sometimes count as persons, too, since (1′) may appear false if only the dog is sleeping; but then perhaps this is a case of a meaning shift to be explained in pragmatic terms. We leave this interesting question open.

[2] One should recall that the notation introduced in D2.3 does not indicate the domain of a function denoted by a lambda-term. In this case it is understood that 'X' stands for predicate extensions.

(10) **Zwei Personen husten.**
[≈ Two persons are coughing.]

As in the case of (1), the focus of our analysis of the subject of (10) is the extension. With respect to the above holiday situation s^* and in analogy to (5), we get the following dependence of the truth value on the predicate extension:

(11) $[\![\textbf{zwei Personen}]\!]^{s^*} =$

$[\![\textbf{husten}]\!]^{s^*}$	0
$[\![\textbf{schlafen}]\!]^{s^*}$	1
$[\![\textbf{sind eine Insel}]\!]^{s^*}$	0
...	...

In (11) the alternatives to **husten** [≈ are coughing] have been listed in their plural form; we will be assuming that this has no impact on their extensions: $[\![\textbf{hustet}]\!]^{s^*}$ [≈ is coughing] = $[\![\textbf{husten}]\!]^{s^*}$ [≈ are coughing], etc. As in the case of **niemand** [≈ nobody] the pattern behind (11) seems easy to make out if one remembers which set is characterized by the predicate extension: $\downarrow [\![\textbf{schlafen}]\!]^{s^*}$ contains two persons, all other sets are free of persons. As a first shot, we may characterize the extension of **zwei Personen** [≈ two persons] as follows:

(12) $[\![\textbf{zwei Personen}]\!]^{s^*} = \lambda X.\vdash\downarrow X \ \cap Per_{s^*}$ contains two elements$\dashv$

The formulation (12) is unclear in that it leaves open what happens if the extension of the predicate (taken as a set) contains *more than two* persons. Does it then contain two persons, in particular, or does it not contain two persons? Does 'two' in (12) mean 'at least two' or does it mean 'exactly two'? In the first case (13b) an adequate account of (12), otherwise (13a) is:

(13) a. $[\![\textbf{zwei Personen}]\!]^{s^*} = \lambda X.\vdash\overline{\overline{\downarrow X \ \cap Per_{s^*}}} = 2\dashv$
b. $[\![\textbf{zwei Personen}]\!]^{s^*} = \lambda X.\vdash\overline{\overline{\downarrow X \ \cap Per_{s^*}}} \geq 2\dashv$

The notation '$\overline{\overline{M}}$ abbreviates – from now on – the *cardinality* of a set M, i.e., the number of its elements.[3] So what is the correct analysis of the noun phrase **two persons**? An adequate answer to this question ought to account for the actual truth values of the relevant sentences. Thus if, in situation s^*, the Müller family are the only Germans around, would then (14) be true or not?

(14) **Zwei Personen sind Deutsche.**
[≈ Two persons are German.]

[3]This only applies to finite sets. The set-theoretic concept of *cardinality* is more general than that of the *number of elements* in that it also applies to infinite sets, for which it was even developed. We need not go into the notion of cardinality for infinite sets though, for which one may consult any textbook on set theory.

As a partial answer to the question of how many persons at the beach are of which nationality, (14) is false – which speaks in favor of interpretation (13a). On the other hand, an Austrian who had just met Horst and Gaby and, apart from them, only seen Englishmen at the beach, may not necessarily be called a liar if he gives (14) as an answer to the question whether there any Germans on the beach at all. This difference in evaluation might indicate an ambiguity in the noun phrase **zwei Personen** [≈ two persons]; but then it may also be due to pragmatic factors. As a matter of fact, this question has not been fully resolved in semantics. Even though (apparently) the majority of semanticists currently tend to favor a pragmatic reading with an underlying literal meaning (13b), alternative analyses have also been defended, according to which, say, the reading (13a) is the underlying one and the other interpretations are due to pragmatics or structural ambiguity. We would like to leave the question open here and only provisionally decide for one of the solutions by making out an ambiguity in the numeral **zwei** [≈ two]; we will return to this in the next section.

From the two alternative extensions given in (13) one may again derive corresponding intensions – simply by abstracting from the given situation s^*, thus passing over to arbitrary situations $s \in LS$:

(15) a. $[\![\textbf{zwei Personen}]\!] = \lambda s.\lambda X.\vdash \overline{\overline{\downarrow X \ \cap Per_s}} = 2 \dashv$

b. $[\![\textbf{zwei Personen}]\!] = \lambda s.\lambda X.\vdash \overline{\overline{\downarrow X \ \cap Per_s}} \geq 2 \dashv$

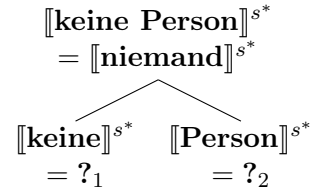
3.2 Determiners

Unlike **zwei Personen** [≈ two persons], the quantified noun phrase **niemand** [≈ nobody] forms an exception in that it consists of a single word[4] and does not decompose into a *determiner* (or article) and a (possibly modified) noun: **zwei** + **Personen**, **eine** + **Person**, **jede** + **Person**, **keine** + **Person**, **die meisten** + **Personen** etc.[5] We will now take a closer look at such 'normal' noun phrases and dissect them into their parts using the abstraction procedure. First of all we must, however, concede that this method does not immediately apply. For though we know the extension of, say, the (entire) expression **keine Person** [≈ no person] in a given situation $s^* \in LS$ – viz. $[\![\textbf{niemand}]\!]^{s^*}$, we have not yet come across the extension of the determiner **kein** [≈ no] or that of the noun **Person** [≈ person]:

[4] There are actually good reasons for assuming that **niemand** [≈ nobody] is not an unstructured expression but consists of three parts, roughly corresponding to the (near) paraphrase **nicht eine Person** [≈ not a person]. Evidence for this *lexical decomposition* of **niemand** [≈ nobody] will be presented in Chapters 5 and 6; in the meantime, we will assume that it is a'monolithic' word.

[5] We assume this bracketing because the structure '**die** + **meisten** + **Personen**' cannot be interpreted in terms of the strategy followed in this chapter.

(16)



The situation visualized in (16) is similar to the starting point of the preceding chapter, where we tried to put together the truth value of predications by combining the extensions of subject and predicate. And we will find a similar way out and determine the noun's contribution to the extension of the whole expression by comparing its alternatives:

(17) a. **Keine Person schläft.**
[≈ No person is sleeping.]
b. **Kein Kind schläft.**
[≈ No child is sleeping.]
c. **Keine Frau schläft.**
[≈ No woman is sleeping.]

Considerations like the one about the Müllers in the above situation s^* show that in the subjects of (17b) and (17c), the rôle Per_{s^*} plays in the extension of **niemand** [≈ nobody] is played, respectively, by the sets Kid_{s^*} and Wom_{s^*} of the children and women in s^*. Thus, e.g., (17b) is true in s^* because only Herr and Frau Müller are asleep and thus the set of kids does not overlap with the extension of the predicate (taken as a set); and since Frau Müller is an element of the intersection of Fra_{s^*} and the predicate extension (taken as a set), (17c) is false in s^*. We thus conclude:

(18) a. $[\![\textbf{Keine Person}]\!]^{s^*} = \lambda X.\vdash\!\downarrow X \ \cap Per_{s^*} = \emptyset\!\dashv$
b. $[\![\textbf{Kein Kind}]\!]^{s^*} = \lambda X.\vdash\!\downarrow X \ \cap Kid_{s^*} = \emptyset\!\dashv$
c. $[\![\textbf{Keine Frau}]\!]^{s^*} = \lambda X.\vdash\!\downarrow X \ \cap Fra_{s^*} = \emptyset\!\dashv$

The contribution a noun N makes to the extension of a quantified (subject) noun phrase of the form '**kein** N' [≈ no N], then, obviously consists in a set of individuals that needs to be disjoint of the predicate extension (taken as a set) if the whole sentence is to come out as true. Since the extensions of noun and predicate each contribute a set of individuals to the whole sentence, it is natural to assimilate them:

(19) a. $[\![\textbf{Person}]\!]^{s^*} = \lambda x.\vdash x \in Per_{s^*}\!\dashv$ $= [\![\textbf{is a Person}]\!]^{s^*}$
b. $[\![\textbf{Kind}]\!]^{s^*} = \lambda x.\vdash x \in Kid_{s^*}\!\dashv$ $= [\![\textbf{is a child}]\!]^{s^*}$
c. $[\![\textbf{Frau}]\!]^{s^*} = \lambda x.\vdash x \in Wom_{s^*}\!\dashv$ $= [\![\textbf{is a woman}]\!]^{s^*}$

According to (19a), any individual u satisfies: $[\![\textbf{Person}]\!]^{s^*}(u) = 1$ iff $u \in Per_{s^*}$. Hence the extension of **Person**[≈ person] (in s^*) characterizes the

set Per_{s^*} of persons (in s^*):

$$\begin{array}{lll} & \downarrow [\![\textbf{Person}]\!]^{s^*} & \\ = & \downarrow \lambda x.\vdash x \in Per_{s^*} \dashv & \text{by (19a)} \\ = & \{x \mid x \in Per_{s^*}\} & \text{by (37), Chapter 2} \\ = & Per_{s^*} & \text{Comprehension Principle} \end{array}$$

And similarly for the other two equations in (19) and the sets Kid_{s^*} and Wom_{s^*}. We could have taken these sets themselves as noun extensions, but then in view of the one-one-relation between sets and their characteristic functions we prefer the parallelism between noun and predicate extensions as defined in (19).

The equations in (19) can, of course, also be formulated without reference to the sets characterized:

(20) a. $[\![\textbf{Person}]\!]^{s^*} = \lambda x.\vdash x$ is a person in $s^* \dashv$
b. $[\![\textbf{Kind}]\!]^{s^*} = \lambda x.\vdash x$ is a child in $s^* \dashv$
c. $[\![\textbf{Frau}]\!]^{s^*} = \lambda x.\vdash x$ is a woman in $s^* \dashv$

(20a) rests on the fact that Per_{s^*} is the set of persons in s^*, so that (any arbitrary) x is a person in s^* just in case $x \in Per_{s^*}$.

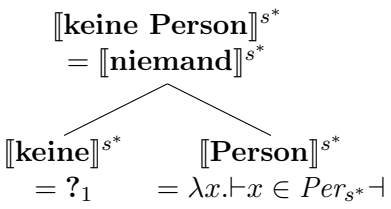
Since obviously nothing in the equations (19) hinges on the peculiarities of the situation s^*, they carry over to arbitrary $s \in LS$; this then fixes the extensions of these nouns:

(21) a. $[\![\textbf{Person}]\!] = \lambda s.\lambda x.\vdash x \in Per_s \dashv$
b. $[\![\textbf{Kind}]\!] = \lambda s.\lambda x.\vdash x \in Kid_s \dashv$
c. $[\![\textbf{Frau}]\!] = \lambda s.\lambda x.\vdash x \in Kid_s \dashv$

So according to (19) and (21) a noun like **Person** [≈ person] has the same extension and intension as the predicate **ist eine Person** [≈ is a person]. This semantic parallelism of nouns and predicates has a long tradition, going back to Aristotle at least, but presumably cannot be upheld in the light of more recent semantic findings. In order to avoid unnecessary complications, we will still apply it here since it turns out to be sufficient for our purposes – the analysis of quantifying noun phrases. It should be noted, though, that not all nouns can be interpreted in this fashion, but only those that can be completed to a full nominal constituent by adding a determinator.[6] Such nouns are also called *sortals*. (21) may thus be taken as a schema for determining the intensions of sortal nouns. Given this schema, the situation

[6] What is excluded are so-called *mass nouns* like **Milch** [≈ milk], which (as a rule) can do without a determiner (and do not form a plural either), as well as so-called *relational* nouns (like **Oberfläche** [≈ surface]), which need a (possibly implicit) complement – ... **des Tischs** [≈ of the table], say. Note that **Kind** [≈ child] and **Frau** [≈ [woman/wife] have relational readings – if they are used in the sense of *direct offspring* and *wife*, respectively – which we are however ignoring here.

dramatically improves vis-à-vis (16):

(22)


(22) describes a typical starting point for an application of the abstraction procedure, according to which the missing extensions of **kein[e]** [≈ no] is now a function that assigns to every noun extension a corresponding quantifier extension. From the observations in (18) and (19) we may first construct the following table:[7]

(23) $[\![\textbf{kein-}]\!]^{s^*} =$

$\lambda x.\vdash x \in Per_{s^*}\dashv$	$\lambda X.\vdash\downarrow X \cap Per_{s^*} = \emptyset\dashv$
$\lambda x.\vdash x \in Kin_{s^*}\dashv$	$\lambda X.\vdash\downarrow X \cap Kid_{s^*} = \emptyset\dashv$
$\lambda x.\vdash x \in Fra_{s^*}\dashv$	$\lambda X.\vdash\downarrow X \cap Wom_{s^*} = \emptyset\dashv$
...	...

The left column of table (23) contains the extensions of our three token nouns according to (19); in the right column they get assigned the corresponding quantifier extensions according to (18). Since the sets Per_{s^*}, Kin_{s^*} and Fra_{s^*} mentioned on the right hand side are just the sets characterized by the noun extensions on the left, the values in the right column may be represented as depending on the corresponding arguments to their left:

(24) $[\![\textbf{kein-}]\!]^{s^*} =$

$[\![\textbf{Person}]\!]^{s^*}$	$\lambda X.\vdash\downarrow X \cap\downarrow [\![\textbf{Person}]\!]^{s^*} = \emptyset\dashv$
$[\![\textbf{Kind}]\!]^{s^*}$	$\lambda X.\vdash\downarrow X \cap\downarrow [\![\textbf{Kind}]\!]^{s^*} = \emptyset\dashv$
$[\![\textbf{Frau}]\!]^{s^*}$	$\lambda X.\vdash\downarrow X \cap\downarrow [\![\textbf{Frau}]\!]^{s^*} = \emptyset\dashv$
...	...
Y	$\lambda X.\vdash\downarrow X \cap\downarrow Y = \emptyset\dashv$
...	...

Unlike table (23), (24) now also contains a typical line that shows the effect of an arbitrary noun extension Y on the extension of the whole quantified noun phrase. Starting from this typical line, the extension of the determiner **kein-** [≈ no] can now be compressed into a lambda-term:

(25) $[\![\textbf{kein-}]\!]^{s^*} = \lambda Y.\lambda X.\vdash\downarrow X \cap\downarrow Y = \emptyset\dashv$

The formula may appear complicated; but it only expresses that the contri-

[7] The hyphen in '**kein-**' [≈ no] is merely supposed to indicate that it is not a particular word form that is at stake but the lexeme with all its forms: **kein** [≈ no] in **kein Mensch** [≈ no human being] has the same meaning (intension) as **keine** [≈ no] in **no person**, etc.

bution the determiner **kein-** [≈ no] makes to the extension of the sentence consists in a condition it imposes after successive application to the extension of noun (Y) and predicate (X) – viz. disjointness (of the sets they characterize).

The extensions of other determiners may now be derived following this model. In each case we only need to know the extensions of quantifying noun phrases they introduce. Let us start with:

(26) **Jede Seminarteilnehmerin hat das Skript gelesen.**
[≈ Every female seminar participant has read the class notes.]

The extension of the subject of (26) is readily identified applying the method of the previous section. If we use 'Sem_{s^*}' to denote the set of female seminar participants in a situation s^*, (26) applies to s^* if every element in this set is an element of the extension of the predicate $[\![\textbf{hat das Skript gelesen}]\!]^{s^*}$ (taken as a set). Varying the predicate again, we find that in general a sentence with **jede Seminarteilnehmerin** [≈ Every female seminar participant] as its subject is true of situations s^* in which every element of Sem_{s^*} appears in the predicate extension (taken as a set). Set-theoretically put, then, for such a sentence to apply to s^*, Sem_{s^*} needs to be a *subset* of the set characterized by the extension of the predicate. In analogy to the equations for quantifying noun phrases of the form '**kein-** N' [≈ '**no** N'] given in (18), we thus arrive at the following analysis:

(27) $[\![\textbf{jede Seminarteilnehmerin}]\!]^{s^*} = \lambda X.\vdash Sem_{s^*} \subseteq \downarrow X \dashv$

Using the extensions of our sample sortals in (19) and (20) we can again determine the contribution of the determiner **jed-** [≈ every] to the quantifier extension in (27) by the abstraction procedure – in close analogy to (24):

(28) $[\![\textbf{jed-}]\!]^{s^*} =$

$[\![\textbf{Person}]\!]^{s^*}$	$\lambda X.\vdash \downarrow [\![\textbf{Person}]\!]^{s^*} \subseteq \downarrow X \dashv$
$[\![\textbf{Kind}]\!]^{s^*}$	$\lambda X.\vdash \downarrow [\![\textbf{Kind}]\!]^{s^*} \subseteq \downarrow X \dashv$
$[\![\textbf{Frau}]\!]^{s^*}$	$\lambda X.\vdash \downarrow [\![\textbf{Frau}]\!]^{s^*} \subseteq \downarrow X \dashv$
$[\![\textbf{Seminarteilnehmerin}]\!]^{s^*}$	$\lambda X.\vdash \downarrow [\![\textbf{Seminarteilnehmerin}]\!]^{s^*} \subseteq \downarrow X \dashv$
...	...
Y	$\lambda X.\vdash \downarrow Y \subseteq \downarrow X \dashv$
...	...

Again, we finally isolate the typical line of table (28) to represent the extension of **jed-** [≈ every] by a lambda-term – as in (25):

(29) $[\![\textbf{jed-}]\!]^{s^*} = \lambda Y.\lambda X.\vdash \downarrow Y \subseteq \downarrow X \dashv$

According to (29) the subset relation plays the same role for **jed-** [≈ every] as does disjointness in the case of **kein-** [≈ no]: if the quantified noun phrase

in question occurs in subject position, the determiner relates the extensions of noun and predicate (taken as sets).

Underlying the extension of the indefinite determiner, there is a simple relation between sets, too.

(30) **Ein Kind schläft.**
[≈ A/one child is sleeping.]

First of all, though, we need to acknowledge that there are several ways of reading (30):

(i) It may express that children in general are asleep – e.g., if (30) is used as an answer to the question **Was macht ein Kind in der Nacht?** [≈ What does a child do at night?.]

(ii) On another, less obvious way of understanding (30), the sentence is false if two or more children are asleep. This is how the sentence comes across if it is used to answer the question **Wie viele Kinder schlafen?** [≈ How many children are asleep?].

(iii) Finally, (30) may just express the opposite of **Kein Kind schläft** [≈ No child is sleeping.] – e.g., if one continues with: ... **und vielleicht schlafen alle** [≈ ... and maybe all of them are.].

In the case of (i) we have a so-called *generic* reading, which we will ignore, because it cannot (or not easily) be captured by our interpretation of quantified noun phrases; we will briefly return to them in Chapter 10 [to be written]. The difference between (ii) and (iii) is that between numeral and indefinite article, as it is expressed in English by **one** vs. **a[n]**. It is, however, not obvious that German also has two distinct words (i.e., readings with the same surface forms) . The situation is reminiscent of the noun phrase **zwei Personen** [≈ two persons] analyzed in the previous section: it may be that the different interpretations of **ein-** [≈ a/one] are due to different uses of one word with one meaning. We will not pursue this question here; for in principle both uses can be accounted for in analogy to **kein-** [≈ no] again. As long as only one of them corresponds to the literal meaning of **ein-** [≈ a/one], it can again be captured by the abstraction procedure. In the case of the indefinite article, this is particularly easy: it merely expresses the opposite of the disjointness expressed by **kein-** [≈ no]:

(31) $[\![\textbf{ein-}_{indef}]\!]^{s^*} = \lambda Y.\lambda X.\vdash\downarrow X \cap \downarrow Y \neq \emptyset\dashv$

The numeral **ein-** [≈ one], too, appears to say something about the relation between noun and predicate extension: on this reading, (30) is true of a situation s^* if the intersection of the extension of **Kind** [≈ child] in s^* – taken as a set $\downarrow [\![\textbf{Kind}]\!]^{s^*}$ – has exactly 1 element – nor more and no less – in common with the extension of **schläft** [≈ sleeps] in s^* – also taken as

a set $\downarrow [\![\textbf{schläft}]\!]^{s^*}$. This obviously neither excludes that there are further children nor that, apart from this one child, any other individuals are asleep. Generalizing from this case, we are thus led to the following analysis of the numeral **ein-** [≈ one]:

(32) a. $[\![\textbf{ein-}_{Num}]\!]^{s^*} = \lambda Y.\lambda X.\vdash\overline{\overline{\downarrow X \cap \downarrow Y}} = 1\dashv$

As we just said, we will leave it open whether the difference between (31) and (32a) is a matter of semantics or whether it is a difference in use. Moreover, unless stated otherwise, if used without subscript, '**ein-**' will, in what follows, be understood as the indefinite article as analyzed in (31).

The other numerals may now be interpreted along the lines of (32a):

(32) b. $[\![\textbf{zwei-}_{Num}]\!]^{s^*} = \lambda Y.\lambda X.\vdash\overline{\overline{\downarrow X \cap \downarrow Y}} = 2\dashv$

c. $[\![\textbf{drei-}_{Num}]\!]^{s^*} = \lambda Y.\lambda X.\vdash\overline{\overline{\downarrow X \cap \downarrow Y}} = 3\dashv$

According to analysis (32b), the above quantifying noun phrase **zwei Personen** [≈ two persons] is interpreted as in (13a); the alternative reading (13b) is obtained by a lexical reading as in (33a), which also generalizes to all other numerals:

(13a) $[\![\textbf{zwei Personen}]\!]^{s^*} = \lambda X.\vdash\overline{\overline{\downarrow X \cap Per_{s^*}}} = 2\dashv$

(13b) $[\![\textbf{zwei Personen}]\!]^{s^*} = \lambda X.\vdash\overline{\overline{\downarrow X \cap Per_{s^*}}} \geq 2\dashv$

(33) a. $[\![\textbf{zwei-}_{Num}]\!]^{s^*} = \lambda Y.\lambda X.\vdash\overline{\overline{\downarrow X \cap \downarrow Y}} \geq 2\dashv$

b. $[\![\textbf{drei-}_{Num}]\!]^{s^*} = \lambda Y.\lambda X.\vdash\overline{\overline{\downarrow X \cap \downarrow X}} \geq 3\dashv$

The disambiguating subscripts used in (33a) and (33b) are supposed to indicate that the extensions are analogous to the extension (31) of the indefinite article. (How so? Exercise!) As already mentioned, we will leave it at this and assume a lexical ambiguity without forgetting that there is something arbitrary about this decision.

In (34), too, the noun extension is related to the predicate extension:

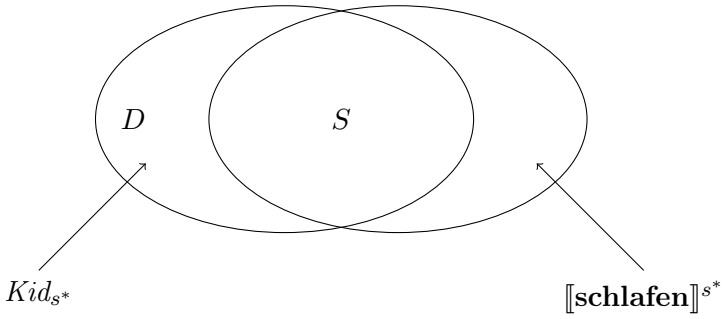
(34) **Die meisten Kinder schlafen.**
[≈ Most children are sleeping.]

Assuming that the plural form **Kinder** [≈ children] has the same meaning as the singular form **Kind** [≈ child],[8] the situations described in (34) can be

[8]This assumption may appear weird: does the plural not make its contribution by allowing for reference to several objects? Matters are not that simple. In a sense, the singular noun **Kind** [≈ child], too, refers to several individuals in that its (changing) extension covers all children (in a situation); without this reference, quantifications like **jedes Kind** [≈ every child] and **kein Kind** [≈ no child] would not be possible. On the other hand, there are in fact uses of plural noun phrases that require reference to *groups* of individuals; the so-called *collective predication* **Fritz und Eike sind [miteinander]**

described as those in which there are more sleeping children than children that are awake. The sleeping children form the intersection ***S*** of the extension of **Kind** [≈ child] and that of **schläft** [≈ is sleeping] (both taken as sets); the children that are awake form the *difference* ***D*** of these sets, i.e., the extension of **Kind** [≈ child] without ('\') the intersection of **schläft** [≈ is asleep]:[9] – graphically:

(35)



The intersection ***S*** obviously needs to contain more elements than ***D*** if (34) is to be true of a situation s^*, i.e., the cardinality of ***S*** must be larger than that of ***D***:

(36) $\overline{\overline{\boldsymbol{S}}} > \overline{\overline{\boldsymbol{D}}}$,
i.e.: $\overline{\overline{Kid_{s^*} \cap \downarrow [\![\textbf{schlafen}]\!]^{s^*}}} > \overline{\overline{Kid_{s^*} \setminus \downarrow [\![\textbf{schlafen}]\!]^{s^*}}}$

In (35) the size of the areas is meant to – exceptionally – represent the relative sizes of the sets represented. (35) thus stands for a typical situation to which (34) applies.

As before we can now glean the extension of the determiner from the concrete case. To do so, we only need to replace the specific noun and predicate extensions mentioned in (36) by variables for arguments to which the extension to be determined can be successively applied. We thus arrive at the following analysis:

(37) $[\![\textbf{die meisten}]\!]^{s^*} = \lambda Y.\lambda X.\vdash \overline{\overline{\downarrow Y \cap \downarrow X}} > \overline{\overline{\downarrow Y \setminus \downarrow X}} \dashv$

We end this semantic tour of the land of determiners with a visit to the

verheiratet [≈ Fritz and Eike are married (to each another)] from Chapter 0 is a case in point. In such cases, plural marking carries meaning. But (34) is not necessarily such a case; for – with the right bracketing – the sentence may be construed as ordinary quantification. Unfortunately, this course will not address the semantics of plural and collectivity. Godehard Link's handbook article 'Plural' (in A. von Stechow, D. Wunderlich (Hrsg.), *Semantik/Semantics*. Berlin 1991, S. 418–440) is recommended as an excellent source.

[9] The difference $A \setminus B$ is defined as the set of elements of the set A that are not at the same time elements of the set B: $A \setminus B := \{x \in A | x \notin B\}$.

definite article **d-** (= **der**/**die**/**das**) [≈ the]. Whether it can be adequately interpreted along the lines of the other determiners is a subject of heavy debate among semanticists. But it is clear that at least certain uses can be captured. Let us look at an example:

(38) **Die türkische Kursteilnehmerin sitzt in der zweiten Reihe.** [≈ The female Turkish course participant is sitting in the second row.]

What situations does s^* (38) apply to? First of all (ii.), in s^* there needs to be a female Turkish course participant who is sitting in the second row. But this alone does not suffice in that (38) cannot really be related to courses in which two or more Turkish women participate: a further condition on s^* is (i.) that there is *precisely one* female Turkish course participant. We thus have, for any situation s^*: $[\![(38)]\!]s^* = 1$ iff the following two conditions are met:

i. there is precisely one female Turkish course participant in s^*;

ii. there is a Turkish course participant in s^* who, in s^*, is sitting in the second row.

Both conditions can again be reformulated as statements about the extensions of noun and predicate, which fits in with a compositional interpretation of the definite article. For i. is about the cardinality of the extension of **türkische Kursteilnehmerin** [≈ female Turkish course participant] (taken as a set); and ii. relates this extension to that of the predicate:

(i.′) $\overline{\overline{\downarrow [\![\textbf{türkische Kursteilnehmerin}]\!]^{s^*}}} = 1$;

(ii.′) $\downarrow [\![\textbf{türkische Kursteilnehmerin}]\!]^{s^*} \cap \downarrow [\![\textbf{sitzt in der 2. Reihe}]\!]^{s^*} \neq \emptyset$

The two conditions are reminiscent of the alternative analyses (31) and (32a) of **ein-** [≈ a/one]. And (ii.′) does indeed give the truth condition of (39) (for situation s^*) if **eine** is interpreted as an indefinite determiner in the sense of (31):

(39) **Eine türkische Kursteilnehmerin sitzt in der zweiten Reihe.** [≈ A female Turkish course participant is sitting in the second row.]

On the other hand (i.′) does *not* match the reading (39) according to which the determiner of the subject is the numeral interpreted as in (32a); for the latter comes with the following truth condition:

(40) $\overline{\overline{\downarrow [\![\textbf{türkische Kursteilnehmerin}]\!]^{s^*} \cap \downarrow [\![\textbf{sitzt in der 2. Reihe}]\!]^{s^*}}} = 1$

Whereas (40) does not exclude that there is more than one female Turkish course participant (as long as only one is sitting in the second row), the condition given in (i′.) says that there is only one female Turkish course participant (no matter where she may be sitting). The truth conditions of (38) thus do not coincide with those of (39), in whichever reading the sentence may be construed.

The conditions (i.′) and (ii.′) can be given in several equivalent ways. For instance, (ii.′) can be reformulated as a condition on the number of elements in the intersection of noun and predicate extension. Alternatively, (ii.′) may be replaced by a subset condition; for (ii.′) boils down to (i.″) in the presence of the condition (i.′), as will be shown in an exercise:

(ii.″) $\quad \downarrow [\![\textbf{türkische Kursteilnehmerin}]\!]^{s^*}$
$\subseteq \quad \downarrow [\![\textbf{sitzt in der zweiten Reihe}]\!]^{s^*}$

(i.′), too, can be reformulated; it may, e.g., be split into two parts:

(i.″) a. $\downarrow [\![\textbf{türkische Kursteilnehmerin}]\!]^{s^*} \neq \emptyset$
b. $\overline{\overline{\downarrow [\![\textbf{türkische Kursteilnehmerin}]\!]^{s^*}}} \leq 1$

(i.″a) says that there is a female Turkish course participant at all, thus guaranteeing that the extension of the noun contains *at least* one element; (i.″b) in turn says that it contains *at most* one element. Together these conditions thus come down to (i.).

Taken together, the conditions (i.″a), (i.″b), and (ii.″) are know as the *Russellian Theory of Descriptions*.[10] More generally, this is the following semantic analysis of the definite article, which is obtained in the by now familiar fashion, by generalizing the sample analysis:

(41) $\quad [\![\textbf{d-}_{Russell}]\!]^{s^*} = \lambda Y.\lambda X.\vdash \underbrace{\downarrow Y \neq \emptyset}_{(i)} \ \& \ \underbrace{\overline{\overline{\downarrow Y}} \leq 1}_{(ii)} \ \& \ \underbrace{\downarrow Y \subseteq \downarrow X}_{(iii)} \dashv$

Condition *(i)*, which corresponds to (i.″a) above, is also called the *existence condition*, because it expresses that there is something – that something exists – in the extension of the noun. The generalization *(ii)* of (i.″b) is called the *uniqueness condition*; for it says that the set $\downarrow Y$ hosts at most one single individual – i.e., not more than one. Condition (iii) does not have a special name.

We will see some of the advantages of this analysis, which at first glance may come across somewhat over-complicated. Even at this point one may object though, that it cannot possible cover all aspects of the use of the

[10] After Bertrand Russell, who in his essay *On Denoting* (1905), opposed an older Fregean analysis with it and at the same time applied it in a critical linguistic analysis of contemporary philosophical works. Russell's analysis had an immense influence on the development of Anglo-Saxon philosophy in the 20$^{\text{th}}$ century.

definite article. The following examples are cases in point:

(42) **Das Auto war zwischen einer Mauer und einem Porsche eingeklemmt.** [≈ The car was clamped between a wall and a Porsche.]
(42) can only apply to situations in which there is more than one car – a Porsche plus the car described by the underlying description. Hence such situations do not meet the uniqueness condition as it is formulated in (41). At best, one could say that the car so-described was the only element in the extension of **Auto** [≈ car] that was *already under discussion.* The uniqueness condition would have to be watered down to a *salience* condition in this case.

(43) **Obwohl Astrid Lindgren gerade in Schweden sehr populär war, hat die Autorin nie den Literaturnobelpreis erhalten.** [≈ Although Astrid Lindgren was particularly popular in Sweden, the author never received the Nobel Prize for literature.]
In (43) the underlined description relates back to the name in the subordinate clause, or as one says in semantics: it is used *anaphorically*; if it is replaced by the pronoun **sie** [≈ she], no detectable difference in meaning ensues. Again it is the uniqueness condition that has been violated: after all, the (main) clause does not relate to situations that contain only one author.

(44) **Der Tiger ist eine in Asien beheimatete Großkatze.** [≈ The tiger is a big cat indigenous to Asia.]
According to the Russellian analysis (41), (44) is supposed to imply that there is only one tiger; but if anything, the uniqueness condition seems to refer to the entire kind *panthera tigris*, not on its exemplars. As in a similar case with indefinites mentioned above, such uses of definite descriptions are called *generic* (in a wide sense).

We will return to these and other objections to the Russellian Theory of Descriptions in Chapter 10 [once its written down], but take it to be correct until then.

For the sake of completeness, we state how the extensions of determiners combine with those of nouns within quantifying noun phrases; since they have been obtained by abstraction, it is again functional application that has to be employed:

(45) *Compositional Determination of the Extension of Quantifying Noun Phrases*
If QN is a quantifying noun phrase consisting of a determiner D and a noun N, the for all $s \in LS$ the following holds:
$[\![QN]\!]^s = [\![D]\!]^s([\![N]\!]^s)$.

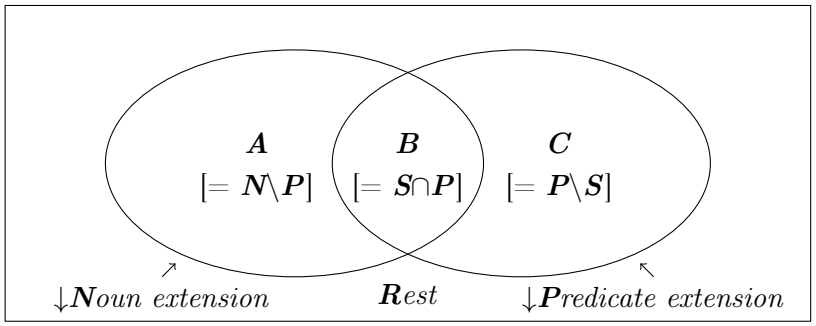
Until now we have focussed exclusively on the extensions of the determiners in a given situation $s^* \in LS$. As always, their intensions emerge once we make the extension depend on an arbitrary situation $s \in LS$:

(46)
a. $[\![\textbf{kein-}]\!] = \lambda s.\lambda Y.\lambda X.\vdash\!\downarrow X \cap \downarrow Y = \emptyset\!\dashv$ cf. (25)
b. $[\![\textbf{jed-}]\!] = \lambda s.\lambda Y.\lambda X.\vdash\!\downarrow X \subseteq \downarrow Y\!\dashv$ cf. (29)
c. $[\![\textbf{ein-}_{indef}]\!] = \lambda s.\lambda Y.\lambda X.\vdash\!\downarrow X \cap \downarrow Y \neq \emptyset\!\dashv$ cf. (31)
d. $[\![\textbf{zwei-}_{indef}]\!] = \lambda s.\lambda Y.\lambda X.\vdash\!\overline{\overline{\downarrow X \cap \downarrow Y}} \geq 2\!\dashv$ cf. (33a)
e. $[\![\textbf{drei-}_{indef}]\!] = \lambda s.\lambda Y.\lambda X.\vdash\!\overline{\overline{\downarrow X \cap \downarrow Y}} \geq 3\!\dashv$ cf. (33b)
f. $[\![\textbf{ein-}_{Num}]\!] = \lambda s.\lambda Y.\lambda X.\vdash\!\overline{\overline{\downarrow X \cap \downarrow Y}} = 1\!\dashv$ cf. (32a)
g. $[\![\textbf{zwei-}_{Num}]\!] = \lambda s.\lambda Y.\lambda X.\vdash\!\overline{\overline{\downarrow X \cap \downarrow Y}} = 2\!\dashv$ cf. (32b)
h. $[\![\textbf{drei-}_{Num}]\!] = \lambda s.\lambda Y.\lambda X.\vdash\!\overline{\overline{\downarrow X \cap \downarrow Y}} = 3\!\dashv$ cf. (32c)
i. $[\![\textbf{die meisten}]\!] = \lambda s.\lambda Y.\lambda X.\vdash\!\overline{\overline{\downarrow Y \cap \downarrow X}} > \overline{\overline{\downarrow Y \setminus \downarrow X}}\!\dashv$ cf. (37)
j. $[\![\textbf{d-}_{Russell}]\!] = \lambda s.\lambda Y.\lambda X.\vdash\!\downarrow Y \neq \emptyset \;\&\; \overline{\overline{\downarrow Y}} \leq 1 \;\&\; \downarrow Y \subseteq \downarrow X\!\dashv$ cf. (43)

3.3 Conservativity and Invariance

The analyses in (46) substantiate that determiners always establish simple set-theoretic relations between noun and predicate extensions that are independent of the situation at hand. In particular, their intensions are always rigid. But the semantic features common to all determiners go deeper. In fact, it turns out that the relations between noun and predicate extensions defined in (46) are not arbitrary but always concern particular 'regions' of the domain of individuals, as partitioned (in a given situation s^*) in the sense of the following Venn diagram:

(47)



It should be noted that the parts called $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, and $\boldsymbol{R}$ in (47) not only depend on the situation s^* at hand but also on the $\boldsymbol{N}$oun and $\boldsymbol{P}$redicate extension; instead we could have written them more clearly but clumsily as: '$\mathbf{A}^{\mathbf{s}^*}_{\mathbf{N,P}}$', '$\mathbf{B}^{\mathbf{s}^*}_{\mathbf{N,P}}$', '$\mathbf{C}^{\mathbf{s}^*}_{\mathbf{N,P}}$', and '$\mathbf{R}^{\mathbf{s}^*}_{\mathbf{N,P}}$'.

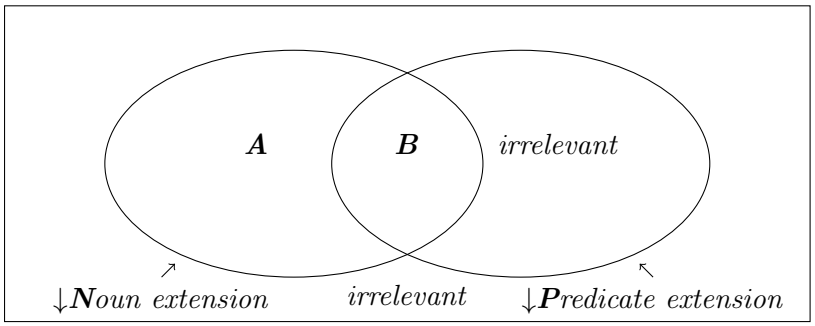
It may be noticed that all of the relations between extensions defined in

(46) only concern the difference $\boldsymbol{A}$ and/or the intersection $\boldsymbol{B}$: the extension of **kein-** [≈ no] says that $\boldsymbol{B}$ is empty; that of **jed-** [≈ every], that $\boldsymbol{A}$ is empty; that of **die meisten** [≈ most], that $\boldsymbol{B}$ contains more elements than $\boldsymbol{A}$; etc.:

(48)	a.	$\mathbf{B} = \emptyset$	**kein-** [≈ no]
	b.	$\mathbf{A} = \emptyset$	**jed-** [≈ every]
	c.	$\mathbf{B} \neq \emptyset$	**ein-**$_{indef}$ [≈ a(n)]
	d.	$\overline{\overline{\mathbf{B}}} \geq 2$	**zwei-**$_{indef}$ [≈ two]
	e.	$\overline{\overline{\mathbf{B}}} \geq 3$	**drei-**$_{indef}$ [≈ three]
	f.	$\overline{\overline{\mathbf{B}}} = 1$	**ein-**$_{Num}$ [≈ one]
	g.	$\overline{\overline{\mathbf{B}}} = 2$	**zwei-**$_{Num}$ [≈ two]
	h.	$\overline{\overline{\mathbf{B}}} = 3$	**drei-**$_{Num}$ [≈ three]
	i.	$\overline{\overline{\mathbf{B}}} > \overline{\overline{\mathbf{A}}}$	**die meisten** [≈ most]
	j.	$\overline{\overline{\mathbf{A}}} = 0 < \overline{\overline{\mathbf{B}}} = 1$	**d-**$_{Russell}$ [≈ the]

It seems as if neither the objects in $\boldsymbol{C}$ nor those in the rest $\boldsymbol{R}$ have a rôle to play in quantification. The extensions of the determiners are, as it were, only interested in two of the four regions:

(47′)



Determiners whose extensions express relations between the regions mentioned in (47′) are called *conservative*. For a conservative determiner only the intersection $\boldsymbol{B}$ and the difference $\boldsymbol{A}$ play a rôle when it comes to determining the truth value. So if a noun and a predicate extension (both taken as sets) have the same difference and the same intersection like another noun extension $\boldsymbol{S'}$ with another predicate extension $\boldsymbol{P'}$, then it does not make a difference to a conservative determiner which of the two noun/predicate combinations it is combined with. We thus arrive at the following definition of conservative determiner extensions:

D3.1 A determinator D is *conservative* if for all situations s and all predicate extensions X, X', Y, and Y' the following holds:

if:	$\downarrow Y \backslash \downarrow X = \downarrow Y' \backslash \downarrow X'$
	and $\downarrow Y \cap \downarrow X = \downarrow Y' \cap \downarrow X'$,
then also:	$[\![D]\!]^s(Y)(X) = [\![D]\!]^s(Y')(X')$

Note that D3.1 is formulated in terms of arbitrary characteristic functions Y, Y', X, and X' rather than the sets $\boldsymbol{S}$, $\boldsymbol{S'}$, $\boldsymbol{P}$, and $\boldsymbol{P'}$; this is so because determiner extensions do not directly apply to sets of indivduals but rather to their characteristic functions.

Intuitively speaking, a conservative determiner makes a statement about the relation between noun and predicate extension to the extent that the latter is affected by the former in the first place: only those objects in the extension of the predicate are taken into account that are also in the extension of the noun. With its extension, the noun marks out the bounds within which the predicate extension is considered. This also explains why statements of the form (a) – with (sortal) noun N and predicate P – can always be reformulated as (b), provided that D is a conservative determiner:

a. $D\ N\ P$
b. $D\ N$ **ist ein(e)** N **und** P
[$\approx$ $D\ N$ is a(n) N and P]

In (a.) the determiner expresses a relation between the noun extension $\downarrow Y$ (taken as a set) and the predicate extension $\downarrow X$ (taken as a set); in (b.) it expresses the same relation between $\downarrow Y$ and $\downarrow Y \cap \downarrow X$ ($= \boldsymbol{B}$). If it is conservative, though, this should come to the same thing; for then it can make a statement about that part of the predicate extension $\boldsymbol{P}$ that overlaps with the noun extension $\boldsymbol{S}$ – in other words, about the intersection $\boldsymbol{B}$. And **Kein Kind schläft** [$\approx$ No child is sleeping] indeed means the same as [$\approx$ No child is a child and is sleeping]; **Jede Frau liebt einen Mann** [$\approx$ Every man loves a woman] means: **Jede Frau ist eine Frau und liebt einen Mann** [$\approx$ Every man is a man and loves a woman]; etc. Given this, a (hypothetical) determiner for which (a.) and (b.) have different truth conditions cannot be conservative. Conversely, any determiner for which (a.) and (b.) coincide, is conservative. We thus obtain the following:[11]

Conservativity Test
A determinator D is conservative just in case, for all situations s and all predicate extensions Y and X, the following holds:
$[\![D]\!]^s(Y)(X) = [\![D]\!]^s(Y)(Y \sqcap X)$,
where $Y \sqcap X$ is the characteristic function of $\downarrow Y \cap \downarrow X$, i.e., $\lambda x.\vdash x \in \downarrow Y \cap \downarrow X \dashv$.

[11] In the literature the characterization given in the Conservativity Test is frequently taken as a definition from which the equivalence in D3.1 is derived. The proof of the equivalence of the two conditions will be provided in an exercise.

The name of the above equivalence derives from the fact that it makes it possible to easily find out whether a determiner is conservative: one only needs to test whether the [schematic] sentences (a.) and (b.) mean the same. As the above examples indicate, this is usually very easy – and certainly easier than relying on the condition given in D3.1 as a test for conservativity.

We use the example of a simple chain of equations for the determiner **kein-** [≈ no] to convince ourselves that the extensions developed above also pass the Conservativity Test:

(49)

	$[\![\textbf{kein-}]\!]^{s^*}(Y)(Y \sqcap X)$	
=	$[\lambda Y.\lambda X.\vdash\downarrow X \cap \downarrow Y = \emptyset\dashv](Y)(Y \sqcap X)$	(25)
=	$[\lambda X.\vdash\downarrow X \cap \downarrow Y = \emptyset\dashv](Y \sqcap X)$	λ-conversion
=	$\vdash \downarrow(Y \sqcap X) \cap \downarrow Y = \emptyset\dashv$	λ-conversion
=	$\vdash \downarrow(\lambda x.\vdash x \in \downarrow Y \cap \downarrow X\dashv) \cap \downarrow Y = \emptyset\dashv$	Def. '$\cap$'
=	$\vdash\{x \mid x \in \downarrow Y \cap \downarrow X\} \cap \downarrow Y = \emptyset\dashv$	(37), Ch. 2
=	$\vdash(\downarrow Y \cap \downarrow X) \cap \downarrow Y = \emptyset\dashv$	Comprehension Principle
=	$\vdash\downarrow Y \cap \downarrow X = \emptyset\dashv$	set theory[12]
=	$[\lambda X.\vdash\downarrow Y \cap \downarrow X = \emptyset\dashv](X)$	λ-conversion
=	$[\lambda Y.\lambda X.\vdash\downarrow Y \cap \downarrow X = \emptyset\dashv](Y)(X)$	λ-conversion
=	$[\![\textbf{kein-}]\!]^{s^*}(Y)(X)$	(25)

The list of the conditions in (48) suggests that all determiners are conservative. This is indeed so – and not just in German [or English, for that matter]: it may well be that the determiners in all languages are conservative. At the same time it is not hard to imagine what a non-conservative determiner would have to look like. Here is an apparent counter-example to universal conservativity:

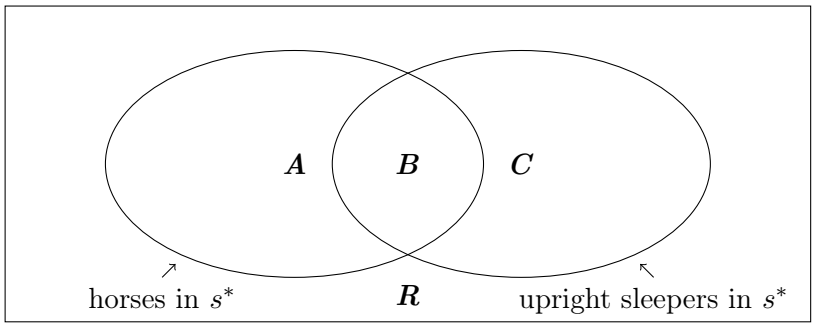
(50) **Nur Pferde können im Stehen schlafen.**
[≈ Only horses can sleep standing up]

(50) is true of a situaiton s^* if there are no individuals in s^* that can sleep standing up without being horses:

[12] This passage ultimately rests on the definition of set-theoretic intersection:

	$A \cap B$	
=	$\{x \mid x \in A \;\&\; x \in B\}$	Def. '$\cap$'
=	$\{x \mid x \in A \;\&\; x \in B \;\&\; x \in B\}$	propositional logic
=	$\{x \mid x \in \{x \mid x \in A \;\&\; x \in B\} \;\&\; x \in B\}$	Comprehension Principle
=	$\{x \mid x \in A \;\&\; x \in B\} \cap B$	Def. '$\cap$'
=	$(A \cap B) \cap B$	Def. '$\cap$'

(51)



In other words, the predicate extension must be a subset of the extension of the noun. This leads to the mirror image of the analysis (29) of **jed-** [≈ every]:

(52) $[\![\textbf{nur}]\!]^{s^*} = \lambda Y.\lambda X.\vdash\!\downarrow X \subseteq \downarrow Y\!\dashv$

According to this analysis, **nur** [≈ only] is not conservative; for the condition on the regions of Logical Space in the partition (51) corresponding to (52) concerns neither $\boldsymbol{A}$ nor $\boldsymbol{B}$, but instead reads:

(53) $\boldsymbol{C} = \emptyset$ **nur** [≈ only]

The above Conservativity Test confirms this finding: if, say, apart from the horses, donkeys also slept standing up, (50) would be false, but the corresponding reformulation (54) would still hold:

(54) **Nur Pferde sind Pferde und können im Stehen schlafen.**
[≈ Only horses are horses and can sleep standing up.]

Is **nur** [≈ only] a non-conservative determiner, then? No – because, even if a casual glance at examples like (54) might suggest otherwise, from a morphosyntactic point of view nothing speaks in favor of **nur** [≈ only] being a determiner in the first place:

- **nur** [≈ only] does not display the kind of agreement morphology typical for German determiners: **viele Kinder** [≈ many_{nom} $\text{children}_{\text{nom}}$] vs. **vielen Kindern** [≈ many_{dat} $\text{children}_{\text{dat}}$], but **nur Kinder(n)** [≈ only $\text{children}_{\text{nom/dat}}$];

- **nur** [≈ only] combines with constituents of all sorts of categories, not just with nouns: **nur schlafen** [≈ only sleep], **nur der König** [≈ only the king], ...; in particular, the construction in (50) can also be construed as modifying the plural indefinite **Pferde** [≈ horses] (and not just the surface-identical noun **Pferde** [≈ horses]);

- Unlike all (other) determiners, **nur** [≈ only] is sensitive to stress dis-

tinctions, even in nominal combinations like (50):[13] **Maria isst nur *Erdbeeren* mit Schlagsahne** [≈ Maria only eats *strawberries* with cream] does not mean the same as **Maria isst nur Erdbeeren mit *Schlagsahne*** [≈ Maria only eats strawberries with *cream*].

Moreover, since **nur** [≈ only] would be the only non-conservative determiner, the quantifier-semantic analysis confirms this diagnosis. We thus note:[14]

(55) *Conservativity Constraint* ... after Barwise & Cooper (1981)
Determiners are always conservative.

Apart from their conservativity, there is a further feature that the conditions various determiners impose on noun and predicate extensions (as listed in (48)), have in common: all of them solely concern the *numbers* of elements of the sets $\boldsymbol{S}$ and $\boldsymbol{P}$. Thus the disjointness condition (46a) expressed by **kein** [≈ no] is satisfied just in case the intersection ($\boldsymbol{B}$) contains zero elements, i.e. if: $\overline{\overline{\mathbf{S} \cap \mathbf{P}}} = 0$. Similarly, the overlap required in (46c) means that $\overline{\overline{\mathbf{S} \cap \mathbf{P}}} \neq 0$ etc. If we abbreviate the cardinalities of $\boldsymbol{A}$ and $\boldsymbol{B}$ as 'a' and 'b', respectively, the conditions in (48) are captured by the following numerical relations:

(56)

a. $b = 0$	**kein-** [≈ no]
b. $a = 0$	**jed-** [≈ every]
c. $b \neq 0$	**ein-**$_{indef}$ [≈ a(n)]
d. $b \geq 2$	**zwei-**$_{indef}$ [≈ two]
e. $b \geq 3$	**drei-**$_{indef}$ [≈ three]
f. $b = 1$	**ein-**$_{Num}$ [≈ one]
g. $b = 2$	**zwei**$_{Num}$ [≈ two]
h. $b = 3$	**drei**$_{Num}$ [≈ three]
i. $b > a$	**die meisten** [≈ most]
j. $a = 0 < b = 1$	**d**$_{Russell}$ [≈ the]

Determiners whose extensions can be reduced to purely arithmetical relations between extensions, along the lines of (56), deserve a term of their own:

D3.2 A determiner D is *invariant* if for all situations s and all predicate extensions X, X', Y, and Y' the following holds:

[13] Though stress distinctions also play a rôle for interpretation elsewhere, in connection with expressions like **nur** [≈ only] they have an effect on the truth conditions. This – well-studied – phenomenon is known as *focus sensitivity*.

[14] Ever since the extremely influential essay 'Generalized Quantifiers and Natural Language' (1981) by Jon Barwise and Robin Cooper, this condition has been regarded as a presumably universally valid semantic constraint.

if: $\overline{\overline{\downarrow Y \backslash \downarrow X}} = \overline{\overline{\downarrow Y' \backslash \downarrow X'}}$,
$\overline{\overline{\downarrow Y \cap \downarrow X}} = \overline{\overline{\downarrow Y' \cap \downarrow X'}}$,
$\overline{\overline{\downarrow X \backslash \downarrow Y}} = \overline{\overline{\downarrow X' \backslash \downarrow Y'}}$, and
$\overline{\overline{U \backslash (\downarrow Y \cup \downarrow X)}} = \overline{\overline{U \backslash (\downarrow Y' \cup \downarrow X'}}$,
then: $[\![D]\!]^s(Y)(X) = [\![D]\!]^s(Y')(X')$

The analyses presented so far suggest that *all* determiners are invariant. But there are remarkable exceptions:

(57) **Peters Auto ist grün.**
[≈ Peter's car is green.]

The possessive **Peters** [≈ Peter's] in (57) plays the rôle of a determiner; and it can be analyzed by a variant of the Russellian Theory of Descriptions applied to the following paraphrase:

(58) **Das Auto, das Peter gehört, ist grün.**
[≈ The car that belongs to Peter is green.]

According to the Russellian analysis, (58) means that (i) Peter owns at least one car, (ii) that he owns at most one car, and that (iii) every car that belongs to Peter is green. Thus the following extension comes out for the subject of (57) and (58):

(59) $[\![\textbf{Peters Auto}]\!]^{s^*} = [\![\textbf{das Auto, das Peter gehört}]\!]^{s^*}$

$= \lambda X. \vdash \downarrow [\![\textbf{Auto, das Peter gehört}]\!]^{s^*} \neq \emptyset$ (i)
$\&$ $\overline{\overline{\downarrow [\![\textbf{Auto, das Peter gehört}]\!]^{s^*}}} \leq 1$ (ii)
$\&$ $\downarrow [\![\textbf{Auto, das Peter gehört}]\!]^{s^*} \subseteq \downarrow X \dashv$ (iii)

$= \lambda X. \vdash \downarrow [\![\textbf{Auto}]\!]^{s^*} \cap PB_{s^*} \neq \emptyset$ (i)
$\&$ $\overline{\overline{\downarrow [\![\textbf{Auto}]\!]^{s^*} \cap PB_{s^*}}} \leq 1$ (ii)
$\&$ $\downarrow [\![\textbf{Auto}]\!]^{s^*} \cap PB_{s^*} \subseteq \downarrow X \dashv$ (iii)

Here 'PB_{s^*}' stands for the objects Peter owns in the situation s^*; for obviously the extension of **Auto, das Peter besitzt** [≈ car that belongs to Peter] results from intersecting the extension of **Auto** [≈ car] (taken as a set) with Peter's belongings.[15] But while, in the case of the subject of (58), the extension in (59) can be obtained by combining the Russellian article with the (complex) noun **Auto, das Peter gehört** [≈ car that belongs to Peter], to get the same result for the subject of (57), the extension of *Peters* [≈ Peter's], which is yet to be determined, needs to be combined with the extension of **Auto** [≈ car]. A quick comparison with pertinent alternative

[15] The precise compositional derivation will be addressed in Chapter 6 in connection with the semantics of relative clauses.

expressions leads to the extension of the possessive in the by now familiar way:[16]

(60) a. $[\![\textbf{Peters Auto}]\!]^{s^*} =$
$\lambda X.\vdash \downarrow [\![\textbf{Auto}]\!]^{s^*} \cap PB_{s*} \neq \emptyset \;\&\; \overline{\overline{\downarrow [\![\textbf{Auto}]\!]^{s^*} \cap PB_{s^*}}} \leq 1 \;\&$
$\downarrow [\![\textbf{Auto}]\!] \cap PB_{s*} \subseteq \downarrow X \dashv$

b. $[\![\textbf{Peters Fahrrad}]\!]^{s^*} =$
$\lambda X.\vdash \downarrow [\![\textbf{Fahrrad}]\!]^{s^*} \cap PB_{s*} \neq \emptyset \;\&\; \overline{\overline{\downarrow [\![\textbf{Fahrrad}]\!]^{s^*} \cap PB_{s^*}}} \leq 1 \;\&$
$\downarrow [\![\textbf{Fahrrad}]\!] \cap PB_{s*} \subseteq \downarrow X \dashv$

c. $[\![\textbf{Peters Haus}]\!]^{s^*} =$
$\lambda X.\vdash \downarrow [\![\textbf{Haus}]\!]^{s^*} \cap PB_{s*} \neq \emptyset \;\&\; \overline{\overline{\downarrow [\![\textbf{Haus}]\!]^{s^*} \cap PB_{s^*}}} \leq 1 \;\&$
$\downarrow [\![\textbf{Haus}]\!] \cap PB_{s*} \subseteq \downarrow X \dashv$

Abstracting from the various noun extensions, we arrive at the following analysis of the possessive:

(61) $[\![\textbf{Peters}]\!]^{s^*} = \lambda Y.\lambda X.\vdash \downarrow Y \cap PB_{s*} \neq \emptyset \;\&\; \overline{\overline{\downarrow Y \cap PB_{s^*}}} \leq 1 \;\&\; \downarrow Y \cap PB_{s*} \subseteq \downarrow X \dashv$

It should be noted that, according to (61), **Peters** [≈ Peter's] is conservative, since in the following reformulation, the sets $\boldsymbol{C}$ and $\boldsymbol{R}$ (in the sense of (47)) play no rôle:

(62) $\overline{\overline{\boldsymbol{A} \cap PB_s}} = 0 \;<\; \overline{\overline{\boldsymbol{B} \cap PB_s}} = 1$ **Peters** [≈ Peter's]

(62) satisfies the conservativity condition even though, on top of $\boldsymbol{A}$ and $\boldsymbol{B}$, it involves ownership; however, the latter is neither contributed by the noun nor by the predicate but part of the meaning of the possessive itself – along with the existence and uniqueness conditions. What is remarkable about (61) and (62) is the fact that these conditions go beyond mere arithmetical relations between noun and predicate extension in that the truth value does not solely depend on the cardinalities (a, b, c, r) of the sections of the domain of individuals according to (47) ($\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{R}$). Thus if (in a given situation) there are exactly as many bicycles as there are cars; exactly as many bicycles made of carbon fibre as there are green cars; and as many green objects as there are objects made of carbon fibre – then the values for a, b, c, and r in (57) and (63) would be the same; but the truth values need not be the same, for in the circumstances, Peter's car could still be green even though

[16] For simplicity, we are ignoring two subtleties. For one thing, the possessive may indicate other relations than *ownership*: thus, e.g., **Peters Haus** [≈ Peter's house] can be the house in which Peter lives. For another thing, the pertinent relation can also be expressed by the noun itself – as in **Peters Vater** [≈ Peter's father], for which no adequate paraphrase **der Vater, der [zu] Peter gehört** [≈ the father who belongs/pertains to Peter] can be found (although this is an additional, remote reading, too). We return to these complications in Chapter 6.

he does not possess a bicycle:

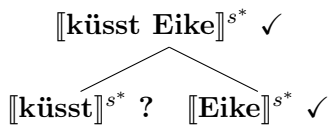
(63) **Peters Fahrrad ist aus Kohlefaser.**
[≈ Peter's bicycle is made of carbon fibre.]

Since the cardinalities a, b, c, and r do not determine the truth value of sentences like (57) and (63), the determiner **Peters** [≈ Peter's] turns out to be not invariant – unlike the other determiners considered here. Still, the uniformity underlying (56) is unlikely to be accidental. For whereas the possessive **Peters** [≈ Peter's] is the result of a grammatical process, the determiners in (56) are all *lexical* – with the possible exception of **die meisten** [≈ most]. And lexical determiners appear to be invariant – not only in German, but perhaps universally so. We thus note:[17]

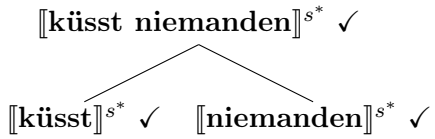
(64) *Logicality Condition*
Lexical Determiners are always invariant.

3.4 Quantifying Objects

Like proper names, quantifying noun phrases do not only occur in subject position. But unlike the former's interpretation, the latter's does not easily carry over from subject to object position. In Section 2.4 we had interpreted the proper name in the predicate **küsst Eike** as if it had been in subject position and then obtained the extension of the predicate by abstraction. So the starting point was:

(65)


For the analysis of a predicate like **küsst niemanden** [≈ is kissing nobody], this strategy is out of the question; for the starting point is different. In the meantime we have identified the extension (and intension) of the transitive verb – precisely by resolving (65). So if we now again assume that its rôle as subject or object makes not difference to the extension of the noun phrase **niemand** [≈ nobody], the abstraction procedure cannot be applied:

(66)


[17] The hypothesis that this, too, is a universal constraint, originates with the essay 'A Semantic Characterization of Natural Language Determiners' (1986) by Edward Keenan and Jonathan Stavi. The authors speak of *logicality* instead of (permutation) invariance, which is the more common term; both terms denote a more general property of extensions that will be scrutinized in the next chapter.

(66) shows that in the case of quantified object noun phrases, the extensions to be combined are all known, and so is the result of the combination. What we do not know, however, is the combination itself, i.e., the operation that merges the extensions of verb and object into the predicate extension. Obviously functional application, which has always been used so far, won't do here, as a quick glance at the extensions to be combined reveals:

(67) a. $[\![\textbf{küsst}]\!]^{s^*} = \lambda y.\lambda x.\vdash x \text{ is kissing } y \text{ in } s^*\dashv$
b. $[\![\textbf{niemand}]\!]^{s^*} = \lambda X.\vdash\!\downarrow X \cap Per_{s^*} = \emptyset\dashv$

In order to be combinable by application, the two extensions need to fit in that one can act as an argument of the other. However, on the one hand, the extension of the verb in (67a) can only be applied to individuals, not to quantifier extensions like $[\![\textbf{niemand}]\!]^{s^*}$. And on the other hand, the latter expects a predicate extension as its argument, i.e., the characteristic function of a set of individuals, whose range is $\{0, 1\}$ (or a subset thereof); but the values of $[\![\textbf{küsst}]\!]^{s^*}$ are no truth values, but predicate extensions themselves. Functional application is thus ruled out as the combination of extensions in (66). Rather, we are looking for a combination $\oplus$ that does the following:

(68) $[\![\textbf{küsst niemanden}]\!]^{s^*}$
$=$ $\lambda x.\vdash x \text{ is kissing nobody in } s^*\dashv$
$=$ $\underline{[\lambda y\lambda x.\vdash\text{in } s^* \ x \text{ is kissing } y\dashv]} \oplus \underline{\underline{[\lambda X.\vdash\!\downarrow X \cap \ Per_{s^*} = \emptyset\dashv]}}$
$=$ $[\![\textbf{küsst}]\!]^{s^*} \oplus [\![\textbf{niemand}]\!]^{s^*}$ – by (38a) from Ch. 2 & (7) from Ch. 3

The problem is to write up the predicate extension in the second line of (68) so that it becomes clear how it results by combining the two extensions in the third line. To this end, we will reformulate the second line until the two extensions have been isolated as separate parts. Once this is done, the result can be generalized to arbitrary quantifying objects.

To begin with, one may observe that the predicate extension (68) can be described in terms of a disjointness conditions as it occurs in the representation (7) of the object extension: that some individual x is kissing nobody (in s^*) just means that the set of persons (in s^*) does not overlap the set of the 'kissees' x (in s^*), the characteristic function of which we will refer to as '$K^x_{s^*}$':

(69) $\lambda x.\vdash x \text{ is kissing nobody in } s^*\dashv = \lambda x.\vdash\!\downarrow K^x_{s^*} \cap Per_{s^*} = \emptyset\dashv$

This account of the predicate extension is reminiscent of the analysis of sentences with quantifying *subjects*, which we developed inSection 3.1:

(70) $[\![\textbf{Niemand wird von Fritz geküsst}]\!]^{s^*}$
$=$ $[\![\textbf{Niemand}]\!]^{s^*}([\![\textbf{wird von Fritz geküsst}]\!]^{s^*})$ by (8)
$=$ $\lambda X.\vdash\!\downarrow X \cap Per_{s^*} = \emptyset\dashv(K^{\text{Fritz}}_{s^*})$ by (7)

$= \quad \vdash\!\downarrow K_{s^*}^{\text{Fritz}} \cap Per_{s^*} = \emptyset\!\dashv$ λ-conversion

The matrix of the second lambda-term in (69) – the condition $\vdash\!\downarrow K_{s^*}^{x} \cap Per_{s^*} = \emptyset\!\dashv$ – looks like the condition of the last line of (70) – with the difference that Fritz has taken over the rôle of the variable x in (69). Now a crucial step in the compositional analysis of the predicate **küsst niemanden** [≈ is kissing nobody] is to transfer the isolation of the quantifier extension $[\![\textbf{niemand}]\!]^{s^*}$ from (70) to the predicate extension (69):

(71) $\lambda x.\vdash\!\downarrow K_{s^*}^{x} \cap Per_{s^*} = \emptyset\!\dashv$
$= \quad [\lambda x.[\lambda X.\vdash\!\downarrow X \cap Per_{s^*} = \emptyset\!\dashv](K_{s^*}^{x})]$ λ-conversion [as in (70)]
$= \quad [\lambda x.[\![\textbf{niemand}]\!]^{s^*}(K_{s^*}^{x})]$ by (7) [as in (70)]

In (70) the argument $K_{s^*}^{\text{Fritz}}$ of $[\![\textbf{niemand}]\!]^{s^*}$ is the extension of the sister constituent **wird von Fritz geküsst** [≈ is being kissed by Fritz]. In (71) the argument $K_{s^*}^{x}$ is, as it were, the extension of the predicate **wird von *x* geküsst** [≈ is being kissed by x] instead.[18]

In order to complete the compositional analysis of the predicate extension, this argument $K_{s^*}^{x}$ still needs to be determined from the extension of the transitive verb **küsst** [≈ is kissing]. This is not particularly hard. $K_{s^*}^{x}$ is the characteristic function of the set of those kissed by x (in s^*) and thus assigns the truth value 1 to an individual y just in case y is being kissed by x; in other words $K_{s^*}^{x}$ assigns to any y the truth value of the statement 'x is kissing y' – the value of $[\![\textbf{küsst}]\!]^{s^*}(y)(x)$. Hence $K_{s^*}^{x}$ can be described easily by the following lambda-term:

(72) $\lambda y.[\![\textbf{küsst}]\!]^{s^*}(y)(x)$

Summarizing the above observations, we thus obtain the following compositional analysis of the extension of the predicate **küsst niemanden** [≈ is kissing nobody]:

(73) $[\![\textbf{küsst niemanden}]\!]^{s^*}$
$= \quad \lambda x.\vdash x \text{ kisses nobody in } s^*\!\dashv$ by (68)
$= \quad \lambda x.\vdash\!\downarrow K_{s^*}^{x} \cap Per_{s^*} = \emptyset\!\dashv$ by (69)
$= \quad \lambda x.[\![\textbf{niemand}]\!]^{s^*}(K_{s^*}^{x})$ by (71)
$= \quad \lambda x.[\![\textbf{niemand}]\!]^{s^*}(\lambda y.[\![\textbf{küsst}]\!]^{s^*}(y)(x))$ by (72)

The decomposition (73) of the extension of the predicate **küsst niemanden** [≈ is kissing nobody] into the extensions of its immediate parts suggests that it can also be used in the general case. We will have ample opportunity to see that this is actually so:

[18] 'as it were', because strictly speaking there is no such predicate, given that German (or English, for that matter) does not contain variables after all. In the seventh chapter [not yet written], we will however introduce an alternative analysis according to which the Logical Forms may contain such variables.

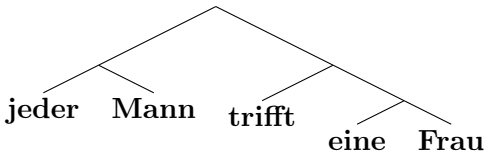
(74) $[\![P]\!]^s = \lambda x.[\![QN]\!]^s(\lambda y.[\![V]\!]^s(y)(x))$

The combination in (74) is clearly more complicated than functional application, which has been used so far for the purpose of composing extensions; it does not have a name of its own but belongs to a family of semantic operations we will get to know better from Chapter 5 onward. For the time being we will content ourselves to test (74) by applying the combination to another example:

(75) **Jeder Mann trifft eine Frau.**
[≈ Every man is meeting a woman.]

We assume the following constituent structure:

(76)



According to the decomposition (76), the immediate parts of (75) are: the quantifying subject **jeder Mann** [≈ every man] and the predicate **trifft eine Frau** [≈ is meeting a woman]. According to the semantics of quantification – the composition rule (8) for quantifying subjects from Section 3.1 – we first get (for any $s^* \in LS$):

(77) $[\![(75)]\!]^{s^*} = [\![\textbf{jeder Mann}]\!]^{s^*}([\![\textbf{trifft eine Frau}]\!]^{s^*})$

Following (47), the extension of the subject is also determined by functional application:

(78) $[\![\textbf{jeder Mann}]\!]^{s^*} = [\![\textbf{jeder}]\!]^{s^*}([\![\textbf{Mann}]\!]^{s^*})$

The two (immediate) constituents of the subject are non-compound expressions; their intension must therefore be listed in the lexicon. According to (46a), we get the extension in (79a) for the determiner **jed-** [≈ every]; for the noun **Mann** [≈ man], we take the equation (79b), which we had been implicitly assuming before.

(79) a. $[\![\textbf{jed-}]\!]^{s^*} = \lambda Y.\lambda X.\vdash\!\downarrow Y \subseteq \downarrow X\!\dashv$
b. $[\![\textbf{Mann}]\!]^{s^*} = \lambda x.\vdash x$ is a man in $s^*\!\dashv$

(77) can now be developed further, where we abbreviate the set of men in s^* by 'Man_{s^*}':

(80) $[\![(75)]\!]^{s^*}$
= $[\![\textbf{jed-}]\!]^{s^*}([\![\textbf{Mann}]\!]^{s^*})([\![\textbf{trifft eine Frau}]\!]^{s^*})$ by (77) and (78)
= $[\lambda Y.\lambda X.\vdash\!\downarrow Y \subseteq \downarrow X\!\dashv]([\![\textbf{Mann}]\!]^{s^*})([\![\textbf{trifft eine Frau}]\!]^{s^*})$ (79a)

$$
\begin{array}{lll}
= & [\lambda X.\vdash\!\downarrow[\![\textbf{Mann}]\!]^{s^*} \subseteq \downarrow X\!\dashv](([\![\textbf{trifft eine Frau}]\!]^{s^*}) & \lambda\text{-conversion} \\
= & [\lambda X.\vdash\!\downarrow(\lambda x.\vdash x \text{ ist a man in } s^*\!\dashv) \subseteq \downarrow X\!\dashv](([\![\textbf{trifft eine Frau}]\!]^{s^*}) & \\
 & & (79b) \\
= & [\lambda X.\vdash\!\{x \mid x \text{ ist a man in } s^*\} \subseteq \downarrow X\!\dashv](([\![\textbf{trifft eine Frau}]\!]^{s^*}) & \\
 & & (37), \text{Ch. 2} \\
= & [\lambda X.\vdash\! Man_{s^*} \subseteq \downarrow X\!\dashv](([\![\textbf{trifft eine Frau}]\!]^{s^*}) & \text{def. of `}Man_{s^*}\text{'} \\
= & \vdash\! Man_{s^*} \subseteq \downarrow[\![\textbf{trifft eine Frau}]\!]^{s^*}\!\dashv & \lambda\text{-conversion}
\end{array}
$$

Before we delve into the analysis of the predicate, we notice some details of the above derivation. In (80) we had performed two λ-conversions to keep the formulae shorter and more transparent (as far as that's possible at all). The exact places at which we performed them do not play a rôle though: as long as one performs all λ-conversions that can be performed at all, the result always – and not just in this case – comes to the same thing.[19] So we could also have performed the lexical insertion according to (79b) and only then get rid of the 'λY':

$$
\begin{array}{lll}
(80') & \ldots & \\
= & [\lambda Y.\lambda X.\vdash\!\downarrow Y \subseteq \downarrow X\!\dashv](([\![\textbf{Mann}]\!]^{s^*})([\![\textbf{trifft eine Frau}]\!]^{s^*}) & (79a) \\
= & [\lambda Y.\lambda X.\vdash\!\downarrow Y \subseteq \downarrow X\!\dashv](\lambda x.\vdash x \text{ ist a man in } s^*\!\dashv)([\![\textbf{trifft eine Frau}]\!]^{s^*}) & \\
 & & (79b) \\
= & [\lambda X.\vdash\!\downarrow(\lambda x.\vdash x \text{ ist a man in } s^*\!\dashv) \subseteq \downarrow X\!\dashv](([\![\textbf{trifft eine Frau}]\!]^{s^*}) & \\
 & & \lambda\text{-conversion} \\
= & \ldots &
\end{array}
$$

However, what we could *not* have done is to get rid of the 'λX' before the 'λY'; for any λ-conversion presupposes a constellation of the form

(81) $[\lambda x. \ldots](a)$

where a is the argument whose place is taken by the variable 'x' in the λ-expression. In (80) we find something of the following form:

(82) $\underline{[\lambda Y.\lambda X. \ldots](a_1)}(a_2)$

In (82) the underlined part is of the form (81); that the variable has a different name is of course unimportant. Hence the 'λY' can indeed be eliminated by λ -conversion as above, plugging in the argument 'a_1' for 'Y' in the part indicated by the three dots. How does one know that a_1 is the argument and not, say, a_2? Quite simply: 'a_1' stands immediately to the right of '$[\lambda Y.\lambda X. \ldots]$'. For the same reason it is just out to first eliminate the 'λX' in (80) by λ-conversion. For in (82) the constellation (81) is found *only in the underlined part*; in particular, neither 'a_1' nor 'a_2' stand immediately

[19] This fact is not immediately obvious, but can be proved with mathematical precision. It is a variant of the so-called *Church-Rosser Theorem* – named after the Church from fn. 9 in Section 2.5 and his pupil John Barkley Rosser who established it in 1936.

to the right of the expression starting with 'λX': the former stands to the right of the 'λY'-term, the latter to the right of the underlined expression. And though (82) does have the overall form '$F(a_2)$', the functional part 'F' is not of the form (81), but is itself composed from '$[\lambda Y.\lambda X.\ldots.]$' and '$a_1$'. Thus in this case the order of the elimination of the lambdas happens to be fixed. So much for the computation (80), which has not even made use of the technique introduced in the current section. This only happens in the next step (where Wom_{s^*} is understood in analogy with Man_{s^*}):

(83)
$[\![\textbf{trifft eine Frau}]\!]^{s^*}$
$= \lambda x.[\![\textbf{eine Frau}]\!]^{s^*}(\lambda y.[\![\textbf{trifft}]\!]^{s^*}(y)(x))$ by (74)
$= \lambda x.[\![\textbf{eine}]\!]^{s^*}([\![\textbf{Frau}]\!]^{s^*})(\lambda y.[\![\textbf{trifft}]\!]^{s^*}(y)(x))$ with (45)
$= \lambda x.[\lambda Y.\lambda X.\vdash\!\downarrow Y \cap \downarrow X \neq \emptyset\!\dashv](([\![\textbf{Frau}]\!]^{s^*})(\lambda y.[\![\textbf{trifft}]\!]^{s^*}(y)(x))$ cf. (31)
$= \lambda x.[\lambda X.\vdash\!\downarrow[\![\textbf{Frau}]\!]^{s^*} \cap \downarrow X \neq \emptyset\!\dashv](\lambda y.[\![\textbf{trifft}]\!]^{s^*}(y)(x))$ λ-conversion
$= \lambda x.\vdash\!\downarrow[\![\textbf{Frau}]\!]^{s^*} \cap \downarrow(\lambda y.[\![\textbf{trifft}]\!]^{s^*}(y)(x)) \neq \emptyset\!\dashv$ λ-conversion
$= \lambda x.\vdash Wom_{s^*} \cap \{y \mid x \text{ is meeting } y \text{ in } s^*\} \neq \emptyset\!\dashv$ (see below)

The final step makes use of the following two lexical equations:

(84) a. $[\![\textbf{Frau}]\!]^{s^*} = \lambda x.\vdash x \text{ is a woman in } s^*\!\dashv\ (= \lambda x.\vdash x \in Wom_{s^*}\!\dashv)$
b. $[\![\textbf{trifft}]\!]^{s^*} = \lambda y.\lambda x.\vdash x \text{ is meeting } y \text{ in } s^*\!\dashv$

(80) and (83) result in the following derivation of the extension of (75):

(85)
$[\![(75)]\!]^{s^*}$
$= \vdash Man_{s^*} \subseteq \downarrow[\![\textbf{trifft eine Frau}]\!]^{s^*}\!\dashv$ by (80)
$= \vdash Man_{s^*} \subseteq \downarrow(\lambda x.\vdash Wom_{s^*} \cap \{y \mid x \text{ is meeting } y \text{ in } s^*\} \neq \emptyset\!\dashv)\!\dashv$ by (83)
$= \vdash Man_{s^*} \subseteq \{x \mid Wom_{s^*} \cap \{y \mid x \text{ is meeting } y \text{ in } s^*\} \neq \emptyset\})\!\dashv$ by (37), Ch. 2

According to (85), the sentence (75) is true of a situation s^* if every element of the set Man_{s^*} – every man in s^* – is an element of the set of x for which the intersection of $\{y \mid x \text{ is meeting } y \text{ in } s^*\}$ and Wom_{s^*} is not empty, i.e., contains at least one element. The last condition means that there is at least one element of Wom_{s^*} – one woman – that at the same time is an element of the set of y that x is meeting. All in all, (75) comes out true of those s^* in which every man is meeting at least one woman. So the combination of extensions for quantifying objects given in (74) also works in this case. We record the following general rule of composition:

(86) *Compositional Determination of the Extension of Object-Quantifications*
If P is a predicate consisting of a transitive verb V and a quantifying noun phrase QN as its object, then for all $s \in LS$ the following holds:

$$[\![P]\!]^s = \lambda x.[\![QN]\!]^s(\lambda y.[\![V]\!]^s(y)(x)).$$

It should be noted that according to this interpretation, a sentence like (87a) may be true of a situation in which every man admires a different actress. This is doubtlessly correct, given that the sentence can indeed be understood in this way. However, couldn't (87a) also mean that every man admires the same actress? This impression is corroborated if the sentence is continued by (87b):

(87) a. **Jeder Mann verehrt eine Schauspielerin.**
[≈ Every man admires an actress.]
b. **Sie hat in unzähligen Filmen mitgespielt.**
[≈ She starred in innumerable films.]

The continuation by (87b) apparently takes it for granted that (87a) reported about a situation in which there is a certain actress who is admired by every man. However, this does not quite prove that we actually have separate reading of the (surface) sentence (87a). For even on an analysis based on (86) would the sentence come out true of such a situation: if an actress is admired by every man, then in particular there is no man who does not admire any actress at all. In order to show that there actually is an ambiguity, one would have to adduce a test, which is to be attempted in the exercises. In the fifth chapter we return to this question and introduce a technique which in principle allows us to represent sentences like (75) or (87a) as ambiguous; pronominal continuations as in (87b) will only be interpreted in the seventh chapter [to be written] though.

3.5 Alternative Interpretations of Quantifying Objects

We end the chapter with two alternative analyses of quantifying object noun phrases. The motivation for this derives from the surprising complexity of the semantic operation used in (86), which gives rise to the question whether the same result could not be obtained in a simpler way. In particular one may wonder if one cannot do with functional application alone – provided that the starting point is suitably modified. This was the starting point at the beginning of the preceding section:

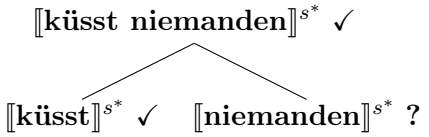
(88)
$$[\![\textbf{küsst niemanden}]\!]^{s^*} = \lambda x.\vdash x \text{ is kissing nobody in } s^* \dashv$$

$$[\![\textbf{küsst}]\!]^{s^*} = \lambda y.\lambda x.\vdash x \text{ is kissing } y \text{ in } s^* \dashv \qquad [\![\textbf{niemand}]\!]^{s^*} = \lambda X.\vdash \downarrow X \cap Per_{s^*} = \emptyset \dashv$$

In view of this scenario, we had been looking for an operation that would combine the known extensions of the parts into the the predicate extension, which was also known. The result can be found in (86). But maybe this procedure was overhasty in that we could as well (or better should) have taken the situation (88) as a reason to revise our previous analyses: if the extensions assumed so far do not combine easily into the predicate extension, then perhaps one needs to assume more complex extensions than had been necessary for the constructions analyzed before. More specifically, then, we may want to probe into the possibility of avoiding the semantic operation (86) by revising *(a)* the predicate extension, or *(b)* the verb extension, or *(c)* the extension of the quantified object. Revision *(a)* would be the most dramatic one because it would have consequences for further, previously analyzed constructions and expressions; the interpretation of quantifying subject noun phrases (and thus the interpretation of determiners) as well as the semantics of predication depended on the predicate extension assumed in (88). We will therefore not pursue the possibility of revising the latter.[20] Neither are we going to speculate whether *(b)* and *(c)* should be modified *at the same time*, with a simpler combination leading to the predicate extension. What we do want to pursue is the possibilities of modifying *either (b) or (c)*.

Starting with *(c)*, we ask ourselves whether the object can be (re-)interpreted so as to have the extension of the predicate **küsst niemanden** [≈ is kissing nobody] fall out of the extension of the verb **küsst** [≈ is kissing] by functional application. Since the object **niemanden**] [≈ nobody$_{acc}$] does not name a particular individual, the possibility of having it as an argument of the verb extension is blocked. So its extension would have to be a function that takes **küsst** [≈ is kissing] as its argument. Which function would that be? The answer again emerges from the abstraction procedure:

(89)



Instead of presenting the solution to (89) in detail, we only record the result, which can be derived in the by now familiar way:

(90) $\quad \llbracket \textbf{niemanden} \rrbracket^{s^*} = \lambda R.\lambda x.\vdash Per_{s^*} \cap \downarrow[\lambda y.R(y)(x)] = \emptyset \dashv$

In (90) the variable 'R' stands for arbitrary extensions of transitive verbs – functions that in turn assign predicate extensions to individuals. As one can easily verify, the application of the extension in (90) to the extension of **küsst** [≈ is kissing] leads to the desired result. What makes this analysis

[20] In connection with the interpretation of so-called *raising verbs* in Chapter 5 we will indicate a revision of the predicate meaning, which however is independent of the problem discussed here.

unsatisfactory, though, is that it forces us to assume that the extension (91) of the subject **niemand** [≈ nobody$_{nom}$] is different from the extension (90) of the object **niemanden** [≈ nobody$_{acc}$].

(91) $[\![\textbf{niemand}]\!]^{s^*} = \lambda X.\vdash\!\downarrow X \cap Per_{s^*} = \emptyset\!\dashv$ [= (7)]

And what is worse: according to this strategy, *every* quantifying noun phrase would have to have different extensions in subject and object position. Moreover, in the case of complex noun phrases like **kein Mensch** [≈ no human being] this duplication of semantic values would carry over to the semantic values of the determiners; for according to our current analysis, the extension of **kein-** [≈ no] combines with that of **Mensch** [≈ human being] into the subject extension (91) and not into the object extension (90). To get from these complications back to systematic analysis, it is now tempting to trace the difference in meaning between accusative and nominative noun phrases to case morphology and derive the extension (90) as in (91):

(92) $[\![\textbf{niemanden}]\!]^{s^*} = [\![\textbf{niemand}_{acc}]\!]^{s^*} = [\![\textbf{niemand}]\!]^{s^*} \oplus [\![accusative]\!]^{s^*}$

In (92) **niemand** (to the left of '$\oplus$') is an uninflected morpheme that gets its extension (91) in the lexicon and can be merged with an accusative-morpheme; the resulting surface form is **niemanden**, whose extension emerges from the morphemes to be combined. This merge is sub-syntactic (part of inflectional morphology), but can still be interpreted compositionally if the extension of the accusative morpheme is determined by abstraction and thus the '$\oplus$' in (92) is taken to stand for functional application. We omit the details again and only show the final result:[21]

(93) a. $[\![accusative]\!]^{s^*} = \lambda F.\lambda R.\lambda x.F(\lambda y R(y)(x))$

b. $[\![accusative]\!]^{s^*}([\![\textbf{niemand}]\!]^{s^*})$ (92)

$= \lambda F.\lambda R.\lambda x.F(\lambda y.R(y)(x))([\![\textbf{niemand}]\!]^{s^*})$ (93a)

$= \lambda F.\lambda R.\lambda x.F(\lambda y.R(y)(x))(\lambda X.\vdash\!\downarrow X \cap Per_{s^*} = \emptyset\!\dashv)$ (7)

$= \lambda R.\lambda x.[\lambda X.\vdash\!\downarrow X \cap Per_{s^*} = \emptyset\!\dashv](\lambda y.R(y)(x))$ λ-conversion

$= \lambda R.\lambda x.\vdash\!\downarrow(\lambda y.R(y)(x)) \cap Per_{s^*} = \emptyset\!\dashv$ λ-conversion

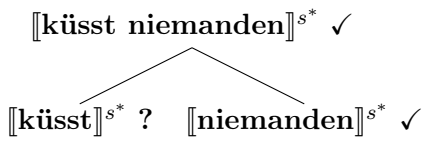
Thus determined, the extension of the accusative morpheme strongly reminds of the complex operation (86) for attaching quantifying objects – which was to be avoided. It now reappears in the guise of a functional morpheme. But this approach, which we will not pursue any further, still manages to do with functional application as the only operation for merging meanings.[22]

[21] The 'F' in (93) stands for extensions of quantifying noun phrases, of course. – For reasons of symmetry one would now expect that the nominative, too, has an extension form which the extension in(91) can be derived by functional application. The following 'void' case meaning achieves this: $[\![nominative]\!]^{s^*} = \lambda F.F$; for $[\lambda F.F]([\![\textbf{niemand}]\!]^{s^*}) = [\![\textbf{niemanden}]\!]^{s^*}$!

[22] Analyses along these lines were popular around 1980 within the tradition of so-called

Another possibility of avoiding operation (86) is by *(b)* revising the semantics of the verb. We thus get the mirror image of the above starting point (89):

(94)



Applying the abstraction procedure to (94) (and again omitting the details), we obtain the following alternative analysis of the verb:

(95) $\llbracket \textbf{küsst} \rrbracket^{s^*} = \lambda F.\lambda x.F(\lambda y.\vdash x \text{ is kissing } y \text{ in } s^* \dashv)$

On this approach too, functional application combines verb extension and object extension, but this time it is the latter that plays the rôle of the argument. Like before, the question arises of how the verb extension (96) assumed so far relates to that in (95):

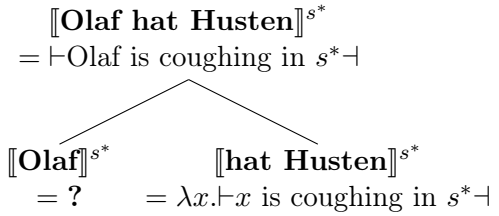
(96) $\llbracket \textbf{küsst} \rrbracket^{s^*} \lambda y.\lambda x.\vdash x \text{ is kissing } y \text{ in } s^* \dashv$

In principle, one may again assume a systematic process that 'raises' ordinary verb extensions as in (96) to more complicated extensions as in (95). However, this process cannot be motivated by inflectional morphology but only by the fact that ordinary verb extensions do not easily combine with quantifying objects, i.e., by functional application. It is therefore the syntactic environment that would have to trigger the process.[23] We will not go into this process in detail but instead present a method of inserting the extension (95) without reducing it to (96). For in contrast to the revision *(c)* of the interpretation of the object, there is no cogent reason for duplicating the verb extension now: so far the only construction transitive verbs undergo is the attachment of proper names as objects, which may also be captured if the transitive verb is interpreted as in (95). One can (but need not) even do without a further semantic operation; instead the extensions of proper names themselves can be revised. Assuming the correctness of our analysis of predicates, we may construct the extensions of names by abstraction in the predication environment:

categorial grammar, but have come out of fashion since they did not fit in well with predominant ideas about syntax.

[23] We will address this process, which is called *type coercion* in semantics, in Section 6.3. It presupposes the central concept of a semantic *type* that we will only encounter in the next chapter but one.

(97)



(97) may appear paradoxical: did we not obtain the above predicate extension by abstraction from the name extension in the first place? How can we now pretend to not know what the extension of the name is? We can. For the abstraction procedure is a heuristics that helps formulating a hypothesis what the meanings of certain expressions might be in cases in which this is unclear. If the hypothesis proves sound when applied to various constructions, it does not matter at the end of the day how it had been obtained. In this sense all analyses are subject to revisions as long as the latter lead to a consistent system of compositional interpretation.

The details of the resolution of the scenario (97) are pushed off to an exercise. In any case, the result is that ⟦**Olaf**⟧$^{s^*}$ is a quantifier extension that directly combines with verb extensions like (95) by functional application. In this way the somewhat baroque lexical equation (95) serves to avoid the complicated semantic operation (86); obviously, then, the complexity is shifted from compositional interpretation to lexical semantics.[24]

Apart from the alternative interpretations of quantifying objects developed or sketched in this chapter there is a further, popular solution to the problem posed at the beginning of the previous section, which makes essential use of an alternative syntactic structure of sentences with quantifying noun phrases – so-called *Quantifier Raising [QR]*. We will get to this approach in [future] Chapter 7.

[24]The interpretation (95) of transitive verbs and the quantifier semantics of proper names that goes with it have been developed by the US-semanticist Richard Montague around 1968; we will address the motivation behind it in Chapters 5 and 6.

3.6 Exercises for Chapter 3

A1 Stepwise derive the extension of (1) in the above holiday situation s^*, in the style of (25‴) of the previous chapter.

A2 Develop the intensions of the following expressions, as is done in the text for **zwei Personen** [≈ two persons]. Proceed by first determining the extensions by abstraction.

i **eine Person** [≈ a person]

ii **jede Person** [≈ every person]

A3 In what sense are the extensions given in (33) generalizations of the extension (31) of the indefinite article?

A4 Show that the semantics of quantification arrived at in this chapter gets the following sentence right:

Kein Mann trifft jede Frau.
[≈ No man is meeting every woman.]

A5 Try to find tests for deciding whether the following sentence is ambiguous:

Jeder Mann verehrt eine Schauspielerin.
[≈ Every man admires an actress.]

A6 What is the extension obtained for the proper name **Olaf** (in a situation s^*) by applying the abstraction procedure to the scenario (97)?

A7 Show that in any situation in which I holds, i and ii come down to the same thing:

I $\overline{\overline{\downarrow [\![\textbf{türkische Kursteilnehmerin}]\!]^{s^*}}} = 1$

i $\downarrow [\![\textbf{türkische Kursteilnehmerin}]\!]^{s^*}$
$\cap \downarrow [\![\textbf{sitzt in der 2. Reihe}]\!]^{s^*} \neq \emptyset$

ii $\downarrow [\![\textbf{türkische Kursteilnehmerin}]\!]^{s^*}$
$\subseteq \downarrow [\![\textbf{sitzt in der 2. Reihe}]\!]^{s^*}$

A8 Show that the concept of conservativity defined in D3.1 implies the correctness of the Conservativity Test.

A9 Try to compositionally account for the *intension* of
Kein Mann heiratet Pippi
[≈ No man is marrying Pipi].

A10 Derive the extensions of the following sentence in the way this is done with the extension of (75) in the text.

I **Drei**$_{Num}$ **Frauen sehen Patrick.**
[≈ Three$_{Num}$ women are seeing Patrick.]

II **Zwei**$_{indef}$ **Kinder schlafen.**
[≈ Two$_{indef}$ children are sleeping.]

III **Der Tiger frisst Roy.**
[≈ The tiger is eating Roy.]

IV **Pippi heiratet keinen Mann.**
[≈ Pipi is marrying no man.]

V **Jedes Kind besitzt einen**$_{indef}$ **Teddy.**
[≈ Every child owns a teddy.]

VI **Der Löwenbändiger schläft.**
[≈ The lion tamer is sleeping.]

VII **Kein Architekt baut ein**$_{Num}$ **Haus.**
[≈ No architect is building one house.]

VIII **Der Postbote beißt den Hund.**
[≈ The postman is biting the dog.]

So that you don't get bored: Pick four of the above sentences.

A11 Describe as accurately as possible what situations must be like in which the last sentence of the previous exercise is not true.

Chapter 4

Intensionality

In the previous three chapters we had seen how various linguistic expressions can be assigned extensions that can then be combined in a systematic, compositional way to form the extensions of ever more complex expressions. In finding these extensions the abstraction procedure proved extremely helpful because in many cases it made possible to identify extensions for expressions for which this seemed to be hardly possible at first, and because the extensions thus determined behaved in a compositional way. In each case we could use the extensions to obtain the corresponding intensions by abstracting from given particular situations, passing to arbitrary possible situations. In this chapter we will look at constructions for which the relation between the extensions of the expressions involved is more complicated. We will mainly focus on one type of expression that has played a major rôle in the history of semantics (and still does): verbs with clause-like complements – or as semanticists call them: *attitude verbs*. We will develop an interpretation for these verbs and the relevant construction (of clausal embedding) that includes elements of logical theories of information processing. At the end of the chapter we will briefly mention further so-called *intensional* constructions which, in this respect, are similar to *attitude reports* (= sentences with attitude verbs) and, for the most part, can be analyzed in a similar way – but not before the end of the next chapter.

4.1 Attitude Reports

This section focuses on verbs that take sentences – and more precisely: **that**-clauses – as objects. Let us start with an example:

(1) **Fritz meint, dass Eike in Berlin ist.**
[≈ Fritz thinks that Eike is in Berlin.]

First of all, we need to observe that the strategy pursued so far – viz. to compositionally reduce the extensions of complex expressions to their im-

mediate parts – fails in this case. To this end we assume (for simplicity) that neither the complementizer **dass** [≈ that] nor the verb-final word order play any semantic rôle.[1] We may than treat (1) and similar examples as if the (verb-second) declarative sentence occupied its object position without having to care about its internal structure. To see that the compositional interpretation fails, we concentrate on the predicate **meint, dass Eike in Berlin ist** [≈ thinks that Eike is in Berlin] and first observe that we are in the typical starting situation to apply the abstraction procedure. Obviously we may take both the embedded sentence and the predicate as having been analyzed, whereas the extension (and the intension) of the attitude verb **meint** [≈ thinks] must still be identified:

(2) **Eike ist in Berlin.** [≈ Eike is in Berlin.]

As always, we are going to consider an arbitrary situation s^* for which the extension of (1) must be determined. If the abstraction procedure were to work in this case, the extension of **meint** [≈ thinks] would have to be a function that assigns the extension of the predicate to the extension of the complement clause (2) – and not just in the case of (1) but also if we are dealing with a different complement clause:

(3) a. $[\![\textbf{meint}]\!]^{s^*}([\![\textbf{Eike ist in Berlin}]\!]^{s^*})$
$=$ $[\![\textbf{meint, dass Eike in Berlin ist}]\!]^{s^*}$
b. $[\![\textbf{meint}]\!]^{s^*}([\![\textbf{Eike ist nicht in Berlin}]\!]^{s^*})$
$=$ $[\![\textbf{meint, dass Eike nicht in Berlin ist}]\!]^{s^*}$

(4) $[\![\textbf{meint, dass Wiesbaden die Hauptstadt Hessens ist}]\!]^{s^*}$
$=$ $[\![\textbf{meint}]\!]^{s^*}([\![\textbf{Wiesbaden ist die Hauptstadt Hessens}]\!]^{s^*})$

Since the complements in (3) and (4) are sentences, their extensions are truth values, and moreover different ones in the variants (3a) and (3b). Given that there are only two truth values, the extension of one of them would have to coincide with the extension of the complement in (4). Thus, e.g., if in s^* Eike is in Frankfurt and (as in real life) Wiesbaden is the capital of Hesse, (5) holds – and hence so does (6):

(5) $[\![\textbf{Eike ist in Berlin}]\!]^{s^*}$
$=$ 0
$=$ $[\![\textbf{Wiesbaden ist nicht die Hauptstadt Hessens}]\!]^{s^*}$

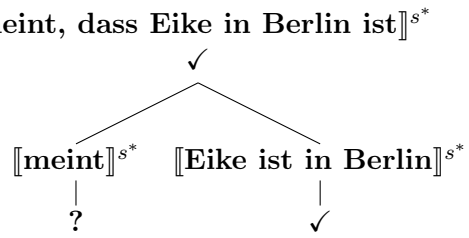
(6) $[\![\textbf{meint}]\!]^{s^*}([\![\textbf{Eike ist in Berlin}]\!]^{s^*})$
$=$ $[\![\textbf{meint}]\!]^{s^*}(0)$
$=$ $[\![\textbf{meint}]\!]^{s^*}([\![\textbf{Wiesbaden ist nicht die Haupstadt Hessens}]\!]^{s^*})$

[1] We could have analyzed all sentences in verb-final position in the first place, deriving the verb-second order by a semantically ineffective surface transformation. In the case at hand, verb-second would also have been possible; but then the following considerations and analyses also apply to attitude verbs that only take verb-final complements.

Yet, however cogent this argument seems, its result is obviously absurd. For according to (6), anybody who is wrong about Eike's whereabouts, would also have to be wrong about the capital of Hesse – and vice versa. And clearly the argument does not depend on the specific example. Quite generally, anyone who has one wrong belief would have to believe everything that is wrong; moreover, a corresponding variation on (5) and (6) would show that everyone who has some accurate belief would be omniscient. Of course, all this is nonsense.

This may look like the end of the theory of extension and intension. Quite the reverse – at least from the historic perspective! In fact, the beginnings of that theory lie precisely in its application to (now) so-called *intensional constructions*, in which the compositionality of extensions fails. In fact, the failure of the analytic strategy followed so far is not due to compositionality as such, but to the fact that it has been applied to *extensions*. After all, the inference from (5) to (6) is based on the assumption that the predicate extension depends on the extensions of its immediate parts, one of which is a sentence – so that any sentence with the same truth value would have to result in the same predicate extension. This *substitution argument*[2] can apparently only be escaped by giving up the assumption that the extension of the predicate is compositionally determined by the extensions of its parts – a hypothesis that originated with the general strategy to reduce the compositionality of intensions to the combination of the simplest possible semantic values, which we had shown at the end of Section 2.6. In view of the substitution argument, one may now try and derive the predicate *intensions* directly from the intensions of the immediate parts. However, it turns out to be sufficient if we just shun the substitution argument 'locally' and only resort to intensions where extensions do not suffice for determining the predicate *extensions* – to wit, for the object clause, which resists substitution by expressions with the same extension. The starting point for determining the predicate extension thereby shifts from (7a) to (7b):[3]

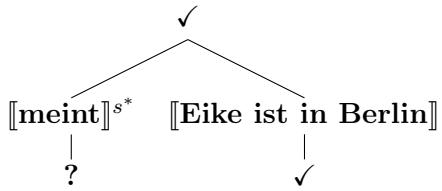
(7) a.



[2]The inference from (5) to (6) to prove the non-compositionality of extensions is a variant of an argument from Gottlob Frege's essay 'Über Sinn und Bedeutung' (1892) [≈ On Sense and Reference].

[3]This 'local' correction of the otherwise compositional semantics of extensions is a crucial characteristic of Frege's analysis of attitude reports.

b. $[\![\textbf{meint, dass Eike in Berlin ist}]\!]^{s^*}$



Both in (7a) and in (7b) the extension of the attitude verb **meint** [≈ thinks] – yet to be determined – is supposed to be a factor in determining the extension of the predicate; the difference is that in (7a) the *extension* of the complement clause is invoked too, whereas in (7b) it is its *intension*. Since the latter is also known, (7b) can in principle again serve as the starting point for an application of the abstraction procedure, according to which the *extension* of **meint** [≈ thinks] is a function that assigns to the *intension* of the complement clause the corresponding *extension* of the predicate:

(8) a. $[\![\textbf{meint, das Eike in Berlin ist}]\!]^{s^*}$
= $[\![\textbf{meint}]\!]^{s^*}([\![\textbf{Eike ist in Berlin}]\!])$
b. $[\![\textbf{meint, dass Eike nicht in Berlin ist}]\!]^{s^*}$
= $[\![\textbf{meint}]\!]^{s^*}([\![\textbf{Eike ist nicht in Berlin}]\!])$
c. $[\![\textbf{meint, dass Wiesbaden die Hauptstadt Hessens ist}]\!]^{s^*}$
= $[\![\textbf{meint}]\!]^{s^*}([\![\textbf{Wiesbaden ist die Hauptstadt Hessens}]\!])$

No absurd consequences like (6) can be derived from the equations in (8). For the above substitution argument depended on the fact that embedded clauses with identical truth values also make the same contribution to the overall extension of the predicate. According to (8), a substitution argument would only allow us to replace two sentences within a predicate if they were *intensionally equivalent* and thus expressed the same proposition. This is not so for the complements in (8), where we obviously have:

(9) $[\![\textbf{Wiesbaden ist die Hauptstadt Hessens}]\!]$
$\neq$ $[\![\textbf{Eike ist in Berlin}]\!]$
$\neq$ $[\![\textbf{Eike ist nicht in Berlin}]\!]$
$\neq$ $[\![\textbf{Wiesbaden ist die Hauptstadt Hessens}]\!]$

And nothing about the relation between the extensions in (8) follows from (9); in particular, they do not have to coincide.[4] The general compositionality rule assumed in (8), then, is:

(10) *Compositional Determination of Clausal Embedding under Attitude*

[4]But then (9) does not *exclude* that they coincide either. However, in the case of (8a) and (8b), this is impossible for independent reasons to be addressed in the next section. For (8a) and (8c), on the other hand, it is quite possible that the extensions coincide; the same is true for (8b) and (8c).

Predicates
If P is a predicate consisting of an attitude verb V and a complement clause S, then for all $s \in LR$ the following holds:
$[\![P]\!]^s = [\![V]\!]^s([\![S]\!])$.

In the spirit of ((7)b), the abstraction procedure can now be used to account for the extension of **meint** [≈ thinks] by way of a table:

(11) $[\![\textbf{meint}]\!]^{s^*} =$

$\lambda s.$ ⊢in s Eike is in Berlin⊣	$\lambda x.$ ⊢in s^*, x thinks that Eike is in Berlin⊣
$\lambda s.$ ⊢in s Eike is not in Berlin⊣	$\lambda x.$ ⊢in s^*, x thinks that Eike is not in Berlin⊣
$\lambda s.$ ⊢in s Wiesbaden is the capital of Hesse⊣	$\lambda x.$ ⊢in s^*, x thinks that Wiesbaden is the capital of Hesse⊣
...	...

The problem is that one would like to know what the '...' in (11) stand for – and in particular, what the typical line looks like. However, it is not clear how the predicate extensions described in the right column of (11) depend on the corresponding (characteristic functions of) the propositions p in the left column – if they do so at all.

(11′) $[\![\textbf{meint}]\!]^{s^*} =$

...	...
p	$\lambda x.$ ⊢ in s^*, x thinks that **???**⊣
...	...

In order to complete (11′), a connection must be made between the situations in which a (given) individual x thinks something and the situations in which whatever x thinks, happens to be the case. To achieve this, one needs a theory that models the beliefs of a person x – that which x thinks is true – in terms of Logical Space. On the basis of such a model, the lacuna in (11′) is easily filled.

4.2 Hintikka semantics

To see how Logical Space can be employed to represent the beliefs of a person, we recall to what situations a statement like (1) (repeated here) applies:

(1) **Fritz meint, dass Eike in Berlin ist.**
[≈ Fritz thinks that Eike is in Berlin.]

To begin with, it is clear that (1) tells us nothing about either *(a)* Eike's

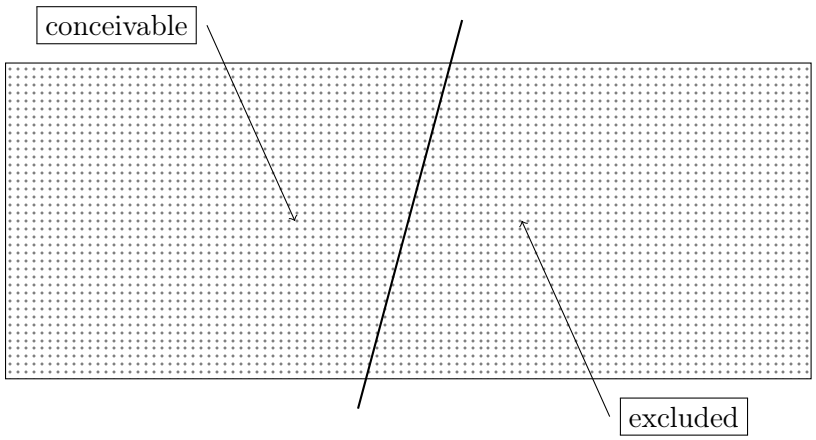
whereabouts or *(b)* Fritz's activities. *(a)* becomes apparent if one compares different scenarios that differ in the truth values of sentence (1) and the clause embedded in it. (This will be done in an exercise.) Concerning *(b)*, we assume that **meinen** [≈ think/remark] is ambiguous (polysemous); here we are only interested in **meinen** as 'thinking', as opposed to the reading of 'remarking'. (The proof of polysemy will also be given in an Exercise.) The sense of **meinen** [≈ think/remark] relevant here does not denote an activity, not even an 'inner' thought-process: (1) can be true after all, even though Fritz is sleeping calmly and dreamlessly. The sentence merely describes a mental constitution of Fritz that manifests itself in his answering 'Berlin' if asked – under certain conditions – for Eike's whereabouts;[5] that – again in certain circumstances – he would display indications of surprise if he were informed that Eike is in Frankfurt; that he put his money on Berlin as the place of Eike's whereabouts; etc. The mental condition attributed in (1) is not a behavior Fritz is actually displaying but merely his potential behavior, a *behavioral disposition*.

According to (1), thus understood, Fritz is in a certain mental state, which we are now going to describe in terms of Logical Space. For this it only matters which beliefs Fritz has in the situation at hand, what is the case according to his belief; it is his *doxastic state* that is a stake.[6] This is all that is relevant in sentence (1); other aspects of Fritz's mental inner life, like the doubts that plague him or his present desires. His beliefs in turn relate to reality and thus to the situation he is in, etc. may be disregarded. Fritz has a certain image of that situation, and that image may be partially correct, partially incorrect, leave out some details, include others. With respect to Logical Space, this means that Fritz's world view is in accordance with some possible situations, but not with others. The latter include all situations in which Eike is in Frankfurt (but not in Berlin), but also all situations in which Germany is a monarchy, in which pigs have wings, etc.; among the former are situations in which Eike is in Berlin, Germany is a republic, and pigs are quadruplets. The distinction between the possible situations in which, according to what he believes, Fritz cannot be, and those that are in accordance with the totality of his beliefs, characterizes Fritz's doxastic state. And – just like the content of a sentence – this distinction can be construed as a bipartition of Logical Space:

[5]The side conditions that must be satisfied are to ensure, among other things, that Fritz's answer is honest (and not a lie or a joke); that he understands and uses the answer correctly (and is not, say, speaking of a different Berlin than the one mentioned in (1)); that putting the question does not influence his belief (perhaps because Eike herself asks him in Tübingen); etc. In the philosophical literature, the diagnosis of beliefs in terms of potential answer behavior under ideal conditions is known as the *disquotational principle*.

[6]The term is supposed to relate to Ancient Greek *doxa* 'belief'.

(12)

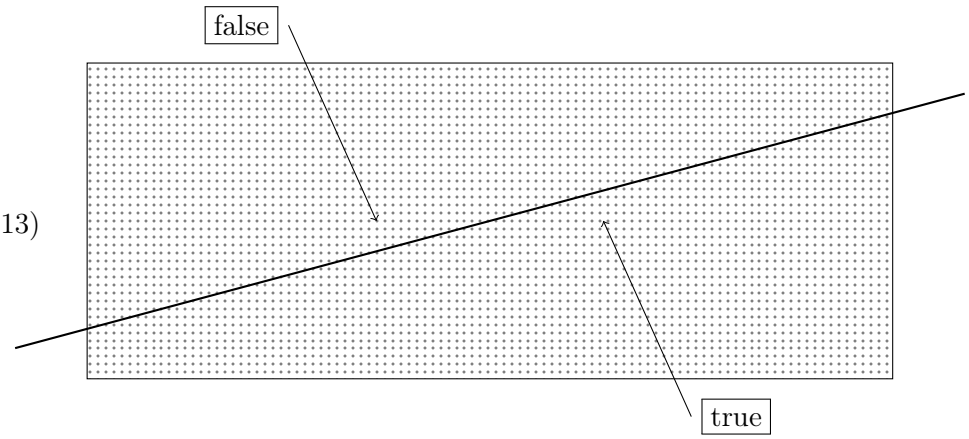


The rectangle in (12) and the dots stand for Logical Space and its elements, the possible situations, respectively. The oblique line symbolizes the cut through Logical Space that Fritz performs by being in the doxastic state he is in: to the right of the line there are the situations that Fritz excludes to be in; all other situations, in which Fritz might be according to his belief, are to the left of the line. The tags *conceivable* and *excluded* are meant to express this. Of course, even situations that Fritz excludes are conceivable for him in that he can *imagine* (being in) them; but he does not *believe* that he is in any of these situations.[7]

The bipartition of Logical Space indicated in (12) matches Fritz's doxastic state in a given situation. The doxastic states of other persons normally match different bipartitions of Logical Space, as do Fritz's doxastic states in different situations. Now, in order to see what it means that (1) is true of a situation, we should recall that the intension of the embedded sentence (2) is also a bipartition of Logical Space, viz. into the situations to which (2) applies, i.e., those of which (2) is *true*, and those of which (2) is *false*:

[7] . . . which means that he believes that he is in none of these situations! Anyone who finds this confusing (rightly so!), should consult the third exercise.

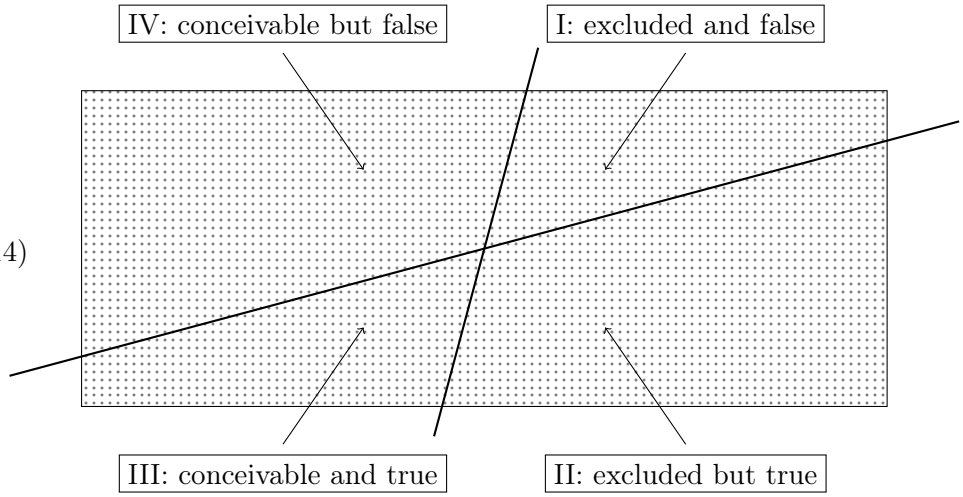
(13)



In the lower situations, it is true that Eike is in Berlin; in the upper ones she is somewhere else (or nowhere). That a situation s is in the lower half thus means that $[\![(2)]\!](s) = 1$; otherwise $[\![(2)]\!](s) = 0$. Since the intension of a sentence is the characteristic function of its content, the lower half in (13) coincides with the content of (2), i.e., $\|(2)\|$

It is understood that the arrangement of the possible situations is the same in (12) and (13). In principle, we are thus dealing with two independent bipartitions of Logical Space. Taken together, Fritz's doxastic state (in the situation at hand) and the content of (2) split up Logical Space into four sectors:

(14)



Sector I contains the situations that Fritz excludes and in which Eike is not in Berlin. Fritz need not exclude the situations *because* in them Eike is not in Berlin, though. If he himself happens to be in Tübingen, say, and is also aware of this, he will, e.g., exclude situations in which he and Eike are both in Berlin – no matter what he thinks about Eike's whereabouts.

The situations of the second sector are those that Fritz excludes and in which Eike is in Berlin. Again Fritz does not have to exclude such situations because in them Eike is in Berlin. He could have different reasons: in some of theses situations, e.g., he may himself be in Moscow even though he knows (or thinks he knows) that he is currently in Tübingen. Fritz would exclude such a situation even if he were convinced that Eike is in Berlin.

Sector Number III consists of the situations in which Eike is in Berlin and to be in which Fritz does not exclude. There are no situations among them of which Fritz can say with certainty that he is in it, but then he cannot exclude this of any of these situations either. They merely differ in aspects about which Fritz does not know enough: in one of them Eike just coughed, in the next one she is asleep, in the third one she is shopping – and all of this happens in Berlin. If (1) is true, there are many such situations; but even if he has no idea or only a vague notion of Eike's whereabouts, this sector will contain many situations – as long as he does not exclude that she is in Berlin.

The fourth and last sector also consists of situations that Fritz cannot exclude to be in; but in none of these situations is Eike in Berlin. If (1) is true, there obviously cannot be such situations; for if Fritz is convinced that Eike is in Berlin, he thereby excludes situations in which she is somewhere else – to wit, all of them. In other words: if (1) is true, Sector IV is empty. And the reverse is also true: if Fritz excludes any situation in which Eike is anywhere but in Berlin, then in any situation that is conceivable for Fritz, Eike is in Berlin, i.e., he takes it that the situation in which he is, is by all means such that in it, Eike is in Berlin.

Whether (1) is true, of course, depends on Fritz's doxastic state and thus on the situation under scrutiny. Following the above considerations, (1) applies to a situation s^* if there is no situation which, given the doxastic state he is in in s^*, Fritz does not exclude and in which Eike is not in Berlin. Positively put this means that (1) is true of s^* just in case in all situations that are conceivable for Fritz in s^*, Eike is in Berlin. So, if Dox_{Fritz,s^*} is the set of situations that Fritz does not exclude in s^* – the 'positive' part of his doxastic state –, the extension of (1) can be characterized as follows, given that the intension is just the characteristic function of the content:

(15) $[\![(1)]\!]^{s^*} = \vdash Dox_{\text{Fritz},s^*} \subseteq \|\textbf{Eike ist in Berlin}\| \dashv$
= $[\![(1)]\!]^{s^*} = \vdash Dox_{\text{Fritz},s^*} \subseteq \downarrow[\![\textbf{Eike ist in Berlin}]\!] \dashv$

On the basis of (15), the compositionality problem posed in (7b) can now be routinely solved. To this end, we first reformulate the extension of the predicate by abstracting from the contribution of the subject:

(16) $[\![\textbf{meint, dass Eike in Berlin ist}]\!]$

$=$ $\lambda x.\vdash x$ thinks that Eike is in Berlin$\dashv$
$=$ $\lambda x.\vdash Dox_{x,s^*} \subseteq \downarrow[\![\textbf{Eike ist in Berlin}]\!]\dashv$

By abstraction from the embedded clause we may then determine the extension of the attitude verb:

(17) $[\![\textbf{meint}]\!]^{s^*} = \lambda p.\lambda x.\vdash Dox_{x,s^*} \subseteq \downarrow p\dashv$

(17) is the core of a semantic analysis of verbs of belief known as *Hintikka semantics*.[8] Other attitude verbs call for other bipartitions of Logical Space but the overall structure of the analysis is always the same. Thus along with the *doxastic perspectives Dox* for the interpretation of knowledge reports, so-called *epistemic perspectives* are employed, and attributions of desires can be reduced to *bouletic perspectives*:[9]

(18) a. $[\![\textbf{weiß, dass Eike in Berlin ist}]\!]$
$=$ $\vdash Epi_{\text{ Fritz},s^*} \subseteq \downarrow[\![\textbf{Eike ist in Berlin}]\!]\dashv$
b. $[\![\textbf{weiß}]\!]^{s^*} = \lambda p.\lambda x.\vdash Epi_{x,s^*} \subseteq \downarrow p\dashv$

(19) a. $[\![\textbf{will, dass Eike in Berlin ist}]\!]$
$=$ $\vdash Bou_{\text{ Fritz},s^*} \subseteq \downarrow[\![\textbf{Eike ist in Berlin}]\!]\dashv$
b. $[\![\textbf{will}]\!]^{s^*} = \lambda p.\lambda x.\vdash Bou_{x,s^*} \subseteq \downarrow p\dashv$

As the doxastic perspective reflects the doxastic state of a person, so their epistemic, bouletic, ... perspectives derive from their epistemic, bouletic, ... states, which themselves correspond to cuts through Logical Space. In a given situation s^*, Fritz does not only distinguish between the situations that he takes to be conceivable ($Dox_{\text{ Fritz},s^*}$) and those that he excludes ($LS\backslash Dox_{\text{ Fritz},s^*}$); but also between those of which he knows that he is not in them ($LS\backslash Epi_{\text{ Fritz},s^*}$) and those of which his knowledge does not enable him to exclude that he is in them ($Epi_{\text{ Fritz},s^*}$); between those in which everything goes according to his desires ($Bou_{\text{ Fritz},s^*}$) and those that in some respect go against them ($LR\backslash Bou_{\text{ Fritz},s^*}$); etc.

None of this excludes that some of these perspectives are related to others or may even be reducible to them. Thus there is a simple connection between the doxastic and the epistemic perspective of a person. If, e.g., Fritz *knows* that he isn't in a particular situation, then his *beliefs* also exclude that he is in that situation. So generally any person x and situation s satisfies: $LS\backslash Epi_{x,s} \subseteq LS\backslash Dox_{x,s}$ – or, positively put:

(20) $Dox_{x,s} \subseteq Epi_{x,s}$

(20) says that a situation in which x might be according to what x thinks

[8] So called after the Finnish logician Jaakko Hintikka, who developed it – on the basis of a number of predecessors – in the 1960s; the standard reference is his essay *Semantics for Propositional Attitudes*, published in 1969.
[9] The terms are supposed to recall Ancient Greek *episteme* 'knowledge' and *boule* 'desire'.

(in s), always is a situation in which x might be according x's knowledge (in s). This elementary connection between knowledge and belief is undisputed. However, it is important to realize that the converse of (20) generally does not hold. For x might be convinced, say, that x has no fever without really knowing this. In that case $Epi_{x,s}$ contains situations in which x has a fever while $Dox_{x,s}$ consists of (x-)fever free situations only. It might also be that x is wrong about his or her state of health: x *thinks* x has no fever, but *as a matter of fact,* x does have a fever. In this case x's doxastic perspective cannot contain the situation x is actually in – for in that situation x does have a fever, and situations in which x has a fever are excluded by x. Quite generally, a person errs (in some respect) in a situation s just in case that person excludes s in the sense that $s \notin Dox_{x,s}$.

Contrary to a possible first impression, (20) thus does not mean at all that the beliefs or convictions of a person x (in a situation s) are always part of x's knowledge. For a belief of x is a proposition $p \subseteq LS$ the truth of x is convinced of – which precisely means, by the Hintikka-semantic analysis, that p is a *superset* of x's doxastic perspective (in s). This, however, does not imply that p is also a superset of x's epistemic perspective; in other words, from $Dox_{x,s} \subseteq p$ and $Dox_{x,s} \subseteq Epi_{x,s}$ it does not follow that: $Epi_{x,s} \subseteq p$. Quite to the contrary: if the latter is the case and thus (again in the spirit of Hintikka semantics) x knows (in s) that p is true, then (20) guarantees that p is also a superset of x's doxastic perspective (in s) and thus x is also convinced of p's truth (in s); in other words, from $Epi_{x,s} \subseteq p$ and $Dox_{x,s} \subseteq Epi_{x,s}$ it follows that $Dox_{x,s} \subseteq p$. These connections will be scrutinized further in the exercises.

Errors in the sense just indicated are not only possible but quite commonplace; so $s \in Dox_{x,s}$ does not always hold. On the other hand, it is impossible that a person x's *epistemic* perspective excludes the situation s that x is in. If this were true, x would have to have knowledge of something that does not apply to x's actual situation s: in s, x would have to know that it is raining although it is not raining in s; or that x has a fever, although x is free from fever in s; etc. But there is no incorrect knowledge: whatever is known is also the case. Epistemic perspectives are thus always subject to the following condition:[10]

(21) $\quad s \in Epi_{x,s}$

(20) and (21) are by no means the only structural properties of doxastic and

[10]The factuality of knowledge (which is part of what semanticists call its *factivity*) stated in (21) is not in conflict with the fact that someone who (honestly) claims to know something, may be wrong; he or she then does not know this but only believes him- or herself to know it. Restrictions on condition (21) are nevertheless possible for situations s in which x does not exist or has no epistemic perspective for some reason or other (perhaps because x is a straw bale). Further modifications of (21) will be made in the ninth chapter [once it has been written].

epistemic perspectives. The relations between belief and knowledge (like those between other attitudes) are not only interesting for lexical semantics; the have been investigated extensively in various sub-areas of philosophy (epistemology, philosophy of mind, philosophical logic). In the [future] seventh chapter we will return to some of the insights gained there.

4.3 The Limits of Hintikka semantics

The Hintikka semantics of attitude reports has its limits. On the one hand, it does not apply to all embeddings of (**that**-) clauses; on the other hand, even in the cases for which it had been developed, it does not always prove to be adequate. For on the one hand, not every attitude report expresses a subset relation, as (17), (18b), and (19b) may suggest; on the other hand, the truth conditions assigned by Hintikka semantics attribute to attitude subjects a higher degree of rationality than can be expected from normal mortal beings. Both problems derive from the crucial fact that according to Hintikka semantics, an attitude report of the form *NN V*, **dass** *S* always implies the truth of *NN V*, **dass** S', provided that S' follows from *S*. Let us recall the observation, made in Section 1.5, that the (sense) relation of implication between sentences corresponds to a subset relation between the propositions they express. We may then describe the crucial fact in the following way:

(22) If $\downarrow[\![S]\!] \subseteq \downarrow[\![S']\!]$, then $\downarrow[\![NN\ V, \textbf{ dass } S]\!] \subseteq \downarrow[\![NN\ V, \textbf{ dass } S']\!]$.

On the Hintikka-interpretation, (22) holds for any attitude verb *V* whose extension (in a situation s^*) is characterized by a suitable perspective *R*:[11]

(23) $[\![V]\!]^{s^*} = \lambda p.\lambda x.\vdash R_{x,s^*} \subseteq \downarrow p \dashv$

As a case in point, (22) implies:

(24) If **Fritz meint, dass Eike auf einer Tagung in Berlin ist**
[≈ Fritz thinks that Eike is at a conference in Berlin]
is true (of some situation), then:
Fritz meint, dass Eike in Berlin oder Benin ist
[≈ Fritz thinks that Eike is in Berlin or Benin]
is true (of that situation).

For we have:

[11] to prove (22) by means of (23), one needs to show that (for any situation s^*) $[\![NN\ V, \textbf{ dass } S']\!]^{s^*} = 1$ as soon as (i) $\downarrow[\![S]\!] \subseteq \downarrow[\![S']\!]$ and (ii) $[\![NN\ V, \textbf{ dass } S]\!]^{s^*} = 1$. But then by (23) and our composition rules we have: $[\![NN\ V, \textbf{ dass } S]\!]^{s^*} = \vdash R_{x,s^*} \subseteq \downarrow[\![S]\!] \dashv$ (where $x = [\![NN]\!]^{s^*}$), i.e., by (i) and (ii): $R_{x,s^*} \subseteq \downarrow[\![S]\!] \subseteq \downarrow[\![S']\!]$ – and thus also: $R_{x,s^*} \subseteq \downarrow[\![S']\!]$, which means that: $\vdash R_{x,s^*} \subseteq \downarrow[\![S']\!] \dashv = 1 = [\![NN\ V, \textbf{ dass } S']\!]^{s^*}$.

(25) $\downarrow[\![\textbf{Eike ist auf einer Tagung in Berlin}]\!]$
$= \{s \in LS \mid \text{in } s, \text{ Eike is at a conference in Berlin}\}$
$\subseteq \{s \in LS \mid \text{in } s, \text{ Eike is in Berlin}\}$
$= \downarrow[\![\textbf{Eike ist in Berlin}]\!]$
$\subseteq \{s \in LS \mid \text{in } s, \text{ Eike is in Berlin or Benin}\}$
$= \downarrow[\![\textbf{Eike ist in Berlin oder in Benin}]\!]$

The implication established in (22), which directly falls out of the Hintikka semantics of **meint** [≈ thinks], is hardly exciting and even welcome in cases like (24): if Fritz thinks that Eike is spending her time at a conference in Berlin, then in particular, he thinks that she is in Berlin – and thus also that she is in Berlin or in Benin.[12] The same goes for attitude reports with **wissen** [≈ know] and many other verbs. Yet in some cases, (22) is obviously wrong. If. e.g., Fritz *excludes* that Eike is at a conference in Berlin, then he must neither exclude that she is in Berlin nor that she is at a conference. The Hintikka-schema (23) cannot be applied to the verb **ausschließen** [≈ exclude] (in the sense as it has been used above, in the introduction of doxastic states). On the other hand, it is clear that in this case too, doxastic perspectives do play a central rôle – just not the same as for **meinen** [≈ think] and **glauben** [≈ believe];[13] the question of which rôle they do play will have to be settled in an exercise.

Apart from the problems with particular attitude verbs, Hintikka semantics suffers from a fundamental defect, which is in part due to modeling information by sets of situations (or is at least amplified by it). In some cases the schema (23) appears to overdo things by taking the attitude holders x to possess more knowledge than they need to have. Here is a case in point:[14]

(26) a. **Fritz meint, dass Wolfgang zwei Töchter hat.**
[≈ Fritz thinks that Wolfgang has two daughters.]
b. **Fritz meint, dass die Anzahl von Wolfgangs Töchtern eine Primzahl ist.**
[≈ Fritz thinks that the number of Wolfgang's daughters is a prime number.]

As a reminder, a prime number is a natural number that is divisible by

[12] This second inference may appear weird: if Fritz is convinced that Eike is in Berlin, one would hardly say that he thinks she is in Berlin *or* Benin; for the latter suggests an insecurity on Fritz's part. This suggestion is commonly explained away as a pragmatic side effect, though; *literally* speaking, Fritz is supposed to think that Eike is in Berlin or Benin as soon as he thinks that she is in one of the two places. Skeptics may try an example without **oder** [≈ or] – perhaps **Fritz meint, dass Eike in Deutschland ist** [≈ Fritz thinks that Eike is in Germany].

[13] We are assuming that, in the relevant readings, **meinen** [≈ think] and **glauben** [≈ believe] are synonymous.

[14] In (26a), **zwei** [≈ two] is meant to be construed in the sense of $\textbf{zwei}_{Num}$ (as defined in Section 3.2).

exactly two numbers. The smallest prime number is 2. This being so, the number of Wolfgang's daughters is prime; for he has two daughters. Since in every possible situation the prime numbers are the same numbers, it also holds of every possible situation in which Wolfgang has two daughters, that the number of his daughters is a prime number. Hence the sentence embedded in (26b) follows from the one embedded in (26a):

(27) $\downarrow[\![\textbf{Wolfgang hat zwei}_{Num}\ \textbf{Töchter}]\!]$
[≈ Wolfgang has two daughters.]
$\subseteq$ $\downarrow[\![\textbf{Die Anzahl von Wolfgangs Töchtern ist eine Primzahl}]\!]$
[≈ The number of Wolfgang's daughters is a prime number.]

According to (22) it would thus follow that $[\![(26a)]\!] \subseteq [\![(26b)]\!]$. But this does not seem to be true: maybe Fritz forgot that 2 is a prime number; maybe he never knew it. In that case, it would seem (26a) could well be the case without (26b) being true.

This argument against Hintikka semantics is based on the assumption that mathematical states of affairs are independent of the situation considered, which makes them somehow trivial: they always hold, no matter where and when, throughout Logical Space. On the other hand, despite their universal truth, mathematical facts are far from being universally known. However if, as in Hintikka semantics, doxastic states are reduced to discriminative abilities in Logical Space, mathematical ignorance obviously cannot be captured. What is true of mathematical states of affairs also holds for facts that pertain due to logic or definition: someone who is convinced that the barber shaves precisely the villagers that do not shave themselves, does not have to believe that the barber is no villager – even though this is logically implied by his conviction; someone who thinks that her dog has arthritis, does not have to know that the animal has problems with its joints, even though this follows from the very definition of *arthritis*. It thus seems as if in cases of lacking *analytic* knowledge (= knowledge based on mathematics, logic, or definition), Hintikka semantics reaches its limits. If we still rely on it here, we will always have to do so under the strong idealization that analytic facts are doxastically trivial.[15]

Apart from attitude reports – or more precisely: clausal embedding under attitude verbs – there is a number of further intensional constructions. The proof of intensionality can usually be given by a *substitution argument*, like the one provided at the beginning of the chapter: if the construction in

[15] A solution to this problem of *logical omniscience* may be conjectured in two areas: for one thing, one may try to work out *models of doxastic (epistemic, ...) states* that are more differentiated than cuts through Logical Space – in terms of sets of propositions perhaps; for another thing, one can try to modify Hintikka's *semantics of attitude reports* by having it not only relate to the intensions of the embedded sentences – but, e.g., also to the intensions of their parts. Proposals along both lines have been made in the philosophical literature, but none of them has proved to be fully convincing so far.

question were extensional, parts with identical extensions could replace each other without any change in the extension of the overall expression. In that sense the following constructions, to which we will come in the next chapter, prove to be intensional, too:

(28) **Fritz will Eike anrufen.**
[≈ Fritz wants to call Eike.] infinitival embedding under *control verbs*

(29) **Fritz sucht ein billiges Restaurant.**
[≈ Fritz is looking for a cheap restaurant.]
object attachment under *opaque verbs*

(30) **Kein Schwein scheint zu grunzen.**
[≈ No pig appears to be grunting.] *raising verbs*

4.4 Exercises for Chapter 4

A1 Show that the truth value of (1) is independent of the truth value of the embedded sentence (2) by describing, for each possible distribution of truth values, a situation to which (1) applies.

A2 Find evidence to the effect that **meinen** in the sense of **think** and **meinen** in the sense of **remark** are distinct expressions.

A3 Judge whether the following sentence pairs are synonymous and, where they are not, describe a situation where the two sentences differ in truth value.

I i **Roy glaubt nicht, dass Siegfried raucht.**
[≈ Roy does not think that Siegfried smokes.]
ii **Roy glaubt, dass Siegfried nicht raucht.**
[≈ Roy thinks that Siegfried does not smoke.]

II i **Siegfried weiß, dass Angela Merkel langsam redet.**
[≈ Siegfried knows that Angela Merkel talks slowly.]
ii **Siegfried weiß, dass die Bundeskanzlerin langsam redet.**
[≈ Siegfried knows that the German chancellor talks slowly.]

A4 The doxastic perspective of a person is a proposition. Contents of sentences are propositions too. Does that mean that doxastic perspectives are contents of sentences?

A5 Show that, according to Hintikka semantics, a person who makes an error, excludes the situation in which he or she is. Does the converse also hold?

Tip: In this context, making an *error* means having (at least) one false belief.

A6 Define the extension of **ausschließen** [≈ exclude], modeling doxastic states by Logical Space.

A7 Find suitable substitution arguments to show that the constructions used in (28)–(30) are intensional.

A8 Consider the sentence:

I **Fritz weiß, dass Eike in Berlin ist.**
[≈ Fritz knows that Eike is in Berlin.]

(a) Show that, according to (17) and (18b), the following implications hold:

i. $\downarrow[\![I]\!] \subseteq \downarrow[\![(1)]\!]$
ii. $\downarrow[\![I]\!] \subseteq \downarrow[\![(2)]\!]$

Make use of the principles (20) and (21).

(b) Refute the following principle (*) by giving a counter-example:

(*) $Epi_{x,s^*} \subseteq Dox_{x,s^*}$.

Chapter 5

Type logic and indirect interpretation

In this chapter we will encounter a systematic method for the representation of extensions and intension that will come in handy in the description of more complex semantic operations in the chapters to come: *type-based indirect interpretation*. The qualification 'indirect' means that the linguistic expressions to be interpreted are translated into an artificial language known as *(two-sorted functional) type logic* – a formal language originating from mathematical logic that has gained the status of a universal auxiliary language in semantics.

5.1 Types

Before we get to the formulae of type logic, we will introduce a classification of the extensions of linguistic expressions that these formulae make reference to: the (*two-sorted functional*) *types*, which are essentially designed to characterize extensions according to whether they are primitive objects or complex functions and if the latter, which kind of complexity they display. So far we have encountered the following kinds of extensions:

(1)

Kind of expression	*Example*	*Kind of extension*
sentence	**Fritz schläft**	truth value
proper name	**Fritz**	individual
intransitive verb/ verb phrase	**schläft**	function from individuals to truth values (= predicate extension)
transitive verb	**küsst**	function from individuals to predicate extensions

ditransitive verb	**übergibt**	function from individuals to functions from individuals to predicate extensions
quantifying noun phrase	**kein Kind**	function from predicate extensions to truth values (= quantifierextension)
(sortal) noun	**Kind**	predicate extension
determiner	**kein**	function from predicate extensions to quantifier extensions
attitude verb	**meint**	function from sentence intensions (= functions from possible situations to truth values) to predicate extensions

With the exception of the first two kinds – truth values and individuals – extensions are always functions. By a function *from As to Bs* we mean a function whose arguments are the *A*s and whose values are always *B*s – where, in the above cases, *A* and *B* are again kinds of extensions – or, as we will say from now on: *types*. In order to systematically account for the above types as well as others to be introduced later, we will, from now on, abbreviate the extension type of 'functions from *A*s to *B*s' as '***(AB)***' and write '***e***' and '***t***' for the *primitive* (= non-functional) types of *indivuduals* and *truth values*, respectively; moreover, we are going to use '***s***' as a label for the set of all possible situations.[1] Given this notation, Table (1) can now be rewritten as in (2) (where we have dropped the outermost brackets around types):

(2)

Kind of expression	*Example*	*Extension type*
sentence	**Fritz schläft**	***t***
proper name	**Fritz**	***e***
intransitive verb/verb phrase	**schläft**	***et***
transitive verb	**küsst**	***e(et)***
ditransitive verb	**übergibt**	***e(e(et))***
quantifying noun phrase	**kein Kind**	***(et)t***
(sortal) noun	**Kind**	***et***
determiner	**kein**	***(et)((et)t)***
attitude verb	**meint**	***(st)(et)***

[1] We thus have: ***s*** = *LS*. The '***e***' is reminiscent of *entity*, which is philosophical jargon for 'object'; '***s***' does not derive from 'situation', but from *sense* (German 'Sinn'), and is supposed to be reminiscent of Frege's term for (his version of) intensions. ***e*** and ***s*** are also called 'sorts'; this term originates from predicate logic, where it denotes 'kinds that cannot be characterised by purely logical means'. The notation for types go back to Montague's *Universal Grammar* (cf. Chapter 0, Fn. 15) and has become standard in semantics.

The only extensions considered so far that do not immediately fit in this pattern are those of coordinating conjunctions, which we had analysed as functions from pairs of truth values to truth values (in Section 1.7): *pairs* of extensions have no place in the above type notation.We could introduce an additional notation for the them; however, we will rather reformulate the analysis of coordinating conjunctions so that they fit in the schema (2). Apart from providing a certain unification, the particular method of reformulation will come in handy for what follows. Let us first recapitulate the analysis of coordination given in Chapter 1:

(3) a. $[\![S \textbf{ und } S']\!]^s = [\![S]\!]^s \times [\![S']\!]^s$ [= (35b) from 1.7]
b. $[\![S \textbf{ oder } S']\!]^s = [\![S]\!]^s + [\![S']\!]^s - [\![S]\!]^s \times [\![S']\!]^s$ [= (36b) from 1.7]

(4) $[\![S\ K\ S']\!]^s = [\![S]\!]^s\ [\![K]\!]^s\ [\![S']\!]^s$ [= (37b) from 1.7]

Given this analysis, the extensions of **und** and **oder** can be construed as functions that take as their arguments two truth values *at once* and assign them a truth value. Such functions can also be represented by way of tables:

(5) a. $[\![\textbf{und}]\!]^s =$

$(0,0)$	0
$(0,1)$	0
$(1,0)$	0
$(1,1)$	1

b. $[\![\textbf{oder}]\!]^s =$

$(0,0)$	0
$(0,1)$	1
$(1,0)$	1
$(1,1)$	1

Unlike the usual truth table representations of the extensions of **und** and **oder** that we met in Chapter 1, the two arguments in (5a) and (5b) only occupy a single column. This difference between the representations is merely cosmetic – the functions represented are the very same, assigning truth values to pairs of truth values. However, using a certain formal hack,[2] functions whose (only) arguments are pairs, may be transformed into complexes of interwoven functions that are applied to the individual components of these pairs, one by one. Replacing *simultaneous* by *successive* application then saves introducing a special type for pairs of truth values (or other extensions). To achieve this, one only needs to modify the extension of the conjunction so that it can first apply to the first and then to the second argument – whereby the result of the first application again needs to be a function, similarly embedded as the ones that we encountered in connection with transitive verbs. Applying this idea to the conjunctions **und** and **oder**, we thus arrive at the following results:

[2]The general procedure is known as *Schönfinkeling* (after the Russian mathematician Moses Schönfinkel [1889 – c. 1942]) or *Currying* (after the US mathematician Haskell Curry [1900 – 1982]) in the literature.

(6) a. $[\![\textbf{und}]\!]^s =$

0	0 0 1 0
1	0 0 1 1

b. $[\![\textbf{oder}]\!]^s =$

0	0 0 1 1
1	0 1 1 1

(6a) and (6b) can now be abbreviated by λ-tems in the familiar fashion:

(7) $[\![\textbf{und}]\!]^s = \lambda v.\lambda u.u \times v$

And for disjunction we get:

(8) $[\![\textbf{oder}]\!]^s = \lambda v.\lambda u.u + v - u \times v$

Though the extensions in (7) and (8) can now be assigned a type in the style of Table (2), the composition rule (4) obviously does not account for them any more; for the result on the right hand-side of the equation ('$[\![S]\!]^s\ [\![K]\!]^s\ [\![S']\!]^{s}$') presupposes that the extension of the conjunction can be simultaneously applied to two arguments. However, it is not hard to adapt (4) to (7) and (8), as will be shown in an exercise.

Let us take stock. Extensions can be classified according to their type, where two kinds of types need to be distinguished: *primitive* and *functional* ones.The type $\boldsymbol{e}$ of individuals, the type $\boldsymbol{t}$ of truth values, and the type $\boldsymbol{s}$ of situations are the primitive types. Functional types always have the form (ab), where a and b are themselves (primitive or functional) types; and the extensions of a functional type (ab) are the functions whose arguments are the extensions of type a and whose values are extensions of type b.

5.2 Type-logical formulae

Like the extensions, the formulae of type logic come in different types, which indicate what sort of object they denote: formulae of type $\boldsymbol{e}$ denote individuals, formulae of type $\boldsymbol{(et)}$ stand for predicate extensions, etc. Accordingly, in indirect interpretation proper names get translated as formulae of type $\boldsymbol{e}$, intransitive verbs translate as formulae of type $\boldsymbol{(et)}$; etc. The translations of complex expressions awill always be composed of the translations of their immediate parts – thereby guaranteeing that the Principle of Compositionality is, as it were, automatically, satisfied in indirect interpretation. As a case in point, a sentence like (9) is translated by suitably combining the translation of the subject **Fritz** with that of the predicate **schläft**:

(9) **Fritz schläft.**

Both parts are lexical expressions that cannot be further syntactically dis-

sembled. Their translations are defined by lexical equations – just like the corresponding meaning. We had assumed the following extensions for the subject and the predicate of (9):

(10) a. $[\![\textbf{Fritz}]\!]^{s^*}$ = Fritz [cf. (5) in 2.1]
b. $[\![\textbf{schläft}]\!]^{s^*} = \lambda x.\vdash x$ is asleep in $s^*\dashv$ [= (2′) in 3.1]

The translations of the lexical expressions are type-logical formulae that reflect essential logical and structural features of the corresponding lexical equations. In equation (10a) the extension of the proper name is defined by making reference to an object; this reference is made by a (homophonic) name of the meta-language, which is not further structured.[3] The type-logical translation of the name **Fritz** will accordingly be a non-compound form, a so-called constant, which will be designated by a boldface letter reminiscent of the word to be interpreted: **f**. Like all type-logical formulae the constant **f** belongs to a type: since it stands for an individual, it is a formula of type $\boldsymbol{e}$. In indirect translation, it is taken for granted that the language of type logic contains enough constants for each type to denote the extensions of the lexical expressions. How many they are and what they look like may be left largely open – as long as any two constants of distinct types are always distinct. A constant is thus always of one type only; this will also hold of type-logical formulae in general.

We could, in principle, also provide a constant for the type-logical translation ot the predicate **schläft**, which would then have to be of type $\boldsymbol{et}$.[4] However, we will instead explicitly express the dependence of the extension on the situation at hand in the translation – as we did in (10b). To do so, we start out with a constant **P** (like **pennt** [≈ 'sleeps', *coll.*]) corresponding to the intension of **schläft**; since the intension assigns predicate extensions to situations, **P** is of type $\boldsymbol{s(et)}$. We then need something to correspond to 's^*' in (10b), i.e. a way to refer to the situation at hand. To this end, we will use the symbol $\boldsymbol{i}$, which however is not a constant but a variable of type $\boldsymbol{s}$ – obviously so, given that it stands for a situation. We will address the difference between constants and variables in due course. One effect of making $\boldsymbol{i}$ a variable is that the formula does not refer to a fixed object.

The type-logical translation of the predicate **schläft** now combines the constant **P** with the variable $\boldsymbol{i}$ so that it refers to the extension of **schläft** (in the situation denoted by $\boldsymbol{i}$). Since this extension is the result of *applying*

[3] *Homophonic* is a sophisticated word for 'sounding alike'. That the object- and meta-linguistic names are pronounced (and written) alike is due to the fact that we analyse German in terms of English, which shares the name. From a theoretical point of view, this is irrelevant but in practice it may occasionally lead to confusion.

[4] This is in line with Montague's approach but would result in logical inconveniences. We will address Montague's method in Section 5.8.

the intension denoted by **P** to the situation denoted by $\boldsymbol{i}$, the whole formula needs to combine the two ingredients so that, as a result, this functional value is denoted. This is achieved by the following construction rule:

(App) If $\boldsymbol{\alpha}$ is a (type-logical) formula of some type ab and $\boldsymbol{\beta}$ is a formula of type a, then $\boldsymbol{\alpha(\beta)}$ is a formula of type b.

$\boldsymbol{\alpha(\beta)}$ is the formula that results by surrounding the formula $\boldsymbol{\beta}$ with (boldface) parentheses and adding the formula $\boldsymbol{\alpha}$ to the left of the whole arrangement; a und b are arbitrary types that may, but need not, be complex. In particular, *(App)* may thus be applied to the constant **P** [for $\boldsymbol{\alpha}$] and the variable $\boldsymbol{i}$ [for $\boldsymbol{\beta}$], given that they are expressions of types $\boldsymbol{s(et)}$ [$= (ab)$] and $\boldsymbol{s}$ [$= a$], respectively. For this case, the rule says that the following formula is of type $\boldsymbol{et}$:

(11) $\mathbf{P(\boldsymbol{i})}$

The label '*(App)*' is of course reminiscent of *[Functional] Application*; for this is how formulae constructed by this rule are supposed to be understood. Strictly speaking, however, *(App)* is only concerned with the *formation* of type-logical formulae, and neither with their meaning nor with their application to indirect interpretation. Though it is clear that a formula like (11) stands for the extension of **schläft** – and formulae of the form $\boldsymbol{\alpha(\beta)}$ generally denote the value of the function denoted by $\boldsymbol{\alpha}$ for the argument denoted by $\boldsymbol{\beta}$; but this does not follow from *(App)* but only from the *interpretation* of type-logical formulae to be introduced in the next section (where the denotations of the constants **f** and **P** will also be officially defined). In a similar vein, it should not come as a surprise that the type-logical translation (12) of sentences like (9) combines the translations of their immediate constituents by *(App)*:

(12) $\mathbf{P(\boldsymbol{i})(f)}$

But again this is not due to *(App)* but will only follow from the translation of natural language into type logic provided by the indirect interpretation procedure to be given in Section 5.5.

The formulae built from constants by applying *(App)* will suffice for the interpretation of the indirect interpretation of many constructions – including, e.g., the direct object in (13):

(13) **trifft Fritz**

If a lexical rule analogous to (11) makes sure that **trifft** translates as a formula of the form $\mathbf{T(\boldsymbol{i})}$ (where this time **T** is a constant of type $\boldsymbol{s(e(et))}$), then the translations of verb and object in (13) can again be combined by *(App)*. As a result, the entire sentence (14) translates into type logic as (15),

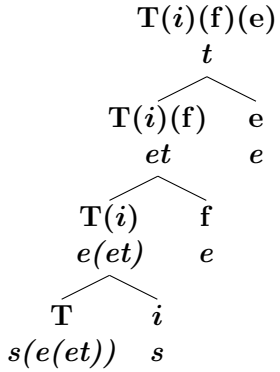
where we take it that **e** is the translation of the name **Eike**):

(14) **Eike trifft Fritz.**

(15) $\underline{\mathbf{T}(\boldsymbol{i})(\mathbf{f})}(\mathbf{e})$

The underlined part of (15) is the translation of the predicate, which ensues by applying *(App)* to $\mathbf{T}(\boldsymbol{i})$ [for $\boldsymbol{\alpha}$] and **f** [for $\boldsymbol{\beta}$]; the formula (15) is obtained once *(App)* combines the underlined predicate translation with the translation of the subject, as in (12). The structure of the formula (15) can be visualised by a tree diagram:

(16)



At each node of the above tree, the second line gives the type of the formula above it. The tree makes clear that *(App)* is a rule of *cancellation* in that the type of a formula constructed by this rule is always shorter than that of its left constituent.[5]

Rule *(App)* meets its limits as a means to express combinations of extensions at the very same point at which direct interpretation got stuck with functional application. In Section 3.4 we saw this happen with the attachment of *quantifying objects*. According to the pertinent rule of combination, the extension of the predicate of (17) was determined by (18):

(17) **Eike trifft niemanden.**

(18) $[\![\textbf{trifft niemanden}]\!]^{s^*} = \lambda x.[\![\textbf{niemand}]\!]^{s^*}(\lambda y.[\![\textbf{trifft}]\!]^{s^*}(y)(x))$

In order to express this combination by a type-logical formula, apart from the possibility *(App)* of referring to function values, one also needs a way of defining functions by (lambda) abstraction. This is precisely what the second construction rule offers:[6]

[5] The analogy to fraction cancellation in arithmetic can be pushed even further by writing types (ab) as $\frac{b}{a}$. *(App)* then combines $\frac{b}{a}$ and a into b – in analogy to multiplication: $\frac{b}{a} \times a = b$.

[6] In order to distinguish them from the variables used in the meta-language, we always refer to variables of type logic by boldface italicized Roman letters. It should be noted that these letters are not themselves the variables but merely meta-linguistic variables

(Abs) If $\boldsymbol{x}$ is a variable of type a and $\boldsymbol{\alpha}$ is a formula of type b, then $\boldsymbol{(\lambda x.\alpha)}$ is a formula of type (ab).

The formula $\boldsymbol{(\lambda x.\alpha)}$ ensues by letting the variable $\boldsymbol{x}$ precede a (boldface) $\boldsymbol{\lambda}$, a (boldface) period, and the formula $\boldsymbol{\alpha}$ and surrounding the arrangement by (boldface) parentheses; a and b are again arbitrary types; $\boldsymbol{\alpha}$ is an arbitrary expression that may, but need not, be complex. *(Abs)* presupposes that the type-logical language contains variables, which are primitive expressions – just like the constants. And like constants, variables are expressions of one – and only one – type. But unlike constants, their number is not left open: we assume that there are *infinitely many of them for each type.* This assumption will turn out to be quite useful in indirect interpretation.[7]

According to *(Abs)* and *(App)* the expressions (19a)–(19c) are type-logical formulae:

(19) a. $\boldsymbol{(\lambda X^{et}.X(\mathbf{f}))}$
b. $\boldsymbol{(\lambda u^{t}.(\lambda v^{t}.(\lambda R^{t(tt)}.R(u)(v))))}$
c. $\boldsymbol{(\lambda x^{e}.\mathbf{T}(i)(\mathbf{f})(x))}$

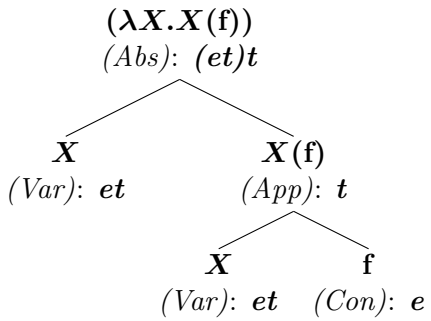
In (19a)–(19c) we followed the convention of indicating the type of a variable as a superscript to its first occurrence in a formula. We will continue to do so in the following; the only exception is made for the variable $\boldsymbol{i}$, which stands for the situation at hand: it will always come without a type superscript.

Once the types of the constants and variables occurring in a formula constructed by *(Abs)* und *(App)* are known, its type is uniquely determined. As a case in point, formula (19a) is of type $\boldsymbol{(et)t}$; for according to *(App)*, $\boldsymbol{X(\mathbf{f})}$ is of type $\boldsymbol{t}$. The structure of the formula can again be represented by way of a tree:

for them: after all, in *(Abs)*, '$\boldsymbol{x}$' does not stand for a specific type-logical variable but for *arbitrary* variables of an equally arbitrary type a. What the variables 'actually' look like will be left open. In connection with type-logical variables we use the common (though not always explicitly mentioned) convention that – unless stated otherwise – within a formula (like $\mathbf{T}(i)(x)(y)$) distinct names of variables ($\boldsymbol{x}$ vs. $\boldsymbol{y}$) stand for distinct variables (whatever they may look like); this does, however, not exclude that distinct variables refer to the very same object (Fritz, say).

[7] A note for hobby mathematicians: 'infinitely many' is not a precise measure. Rather, in mathematics, several infinite magnitudes are distinguished. It is usually assumed that there are *countably* many variables (for each type and all in all) – which is relatively few by infinity standards. (By way of comparison: the natural numbers are countable, the reals are not.)

(20)



The above tree also indicates which rule was used to construct the parts it is composed of; this piece of information precedes the type. Hence the second line of the top node says: according to *(Abs)*, the formula in the line above is of type ***(et)t***. In the case of primitive, non-compound formulae, instead of a construction rule, the *kind* of expression is indicated, i.e. whether it is a variable or a constant. Using (20) as a model, the reader can now determine the types of both of the formulae (19b) and (19c), which is done in an exercise.

(App) und *(Abs)* are the only construction rules of type logic: all formulae are constructed from constants and variables using these two operations. We thus arrive at the following:

(21) *Syntax of (two-sorted functional) type logic*:
For all types a and b, the following holds:

(Var) Variables of type a are formulae of type a.

(Con) Constants of type a are formulae of type a.

(App) If $\boldsymbol{\alpha}$ is a (type-logical) formula of a type (ab) and $\boldsymbol{\beta}$ is a formula of type a, then $\boldsymbol{\alpha(\beta)}$ is a formula of type b.

(Abs) If $\boldsymbol{x}$ is a variable of type a and $\boldsymbol{\alpha}$ is a formula of type b, then $\boldsymbol{(\lambda x.\alpha)}$ is a formula of type (ab).

(21) is to be understood as implying that nothing counts as a type-logical formula of a type a, unless it has been constructed by one of these rules. (21) finishes the definition proper of type-logical formulae. However, before we get to their interpretation – for (21) only says how the formulae are structured, not what they stand for – we will introduce four so-called *logical* constants, which play a central rôle in the practice of indirect interpretation, because they are subject to certain *rules of calculation* that allow for a quasi-mechanical reduction of confusingly complex formulae.

The first logical constant is a special symbol for clausal conjunction (in the semantic sense), which – as we have seen in the previous section – can be 'mimicked' by a function of type ***t(tt)***. In type logic this function is denoted by a constant of type ***t(tt)***, for which we will borrow the symbol $\wedge$ from

propositional logic.[8] In analogy to natural language, this symbol is inserted between the conjoined formulae; thus if $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ are formulae of type $\boldsymbol{t}$, instead of $\wedge(\boldsymbol{\psi})(\boldsymbol{\varphi})$, we will write $[\boldsymbol{\varphi} \wedge \boldsymbol{\psi}]$, which reads '$\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$'.

The constant $\wedge$ is not only used to analyze the word **und**; it plays an important rôle in indirect interpretation and in type logic in general. In particular, it forms the core of the semantics of the modifier constructions (relative clauses, adjectives) to be addressed in the next chapters; and it will already be applied in several lexical analyses in the current chapter. The translation of the indefinite article is a case in point. In the preceding chapter we analysed it as follows:

(22) $\quad [\![\mathbf{ein}_{indef}]\!]^{s^*} = \lambda Y.\lambda X.\vdash\!\downarrow X \cap \downarrow Y \neq \emptyset\!\dashv$ $\quad$ [= (31) from 3.2]

Together with a further constant, the conjunction $\wedge$ will help us to represent the extension to the right of the equality sign in (22) within type logic. To see how this works, let us first slightly reformulate the equation:

(23) $\quad [\![\mathbf{ein}_{indef}]\!]^{s^*} = \lambda Y.\lambda X.\vdash\!\downarrow[\lambda x.X(x) \times Y(x)] \neq \emptyset\!\dashv$

Where (22) mentions the intersection of the sets characertised by X and Y, (23) has the set characterised by the lambda-term '$\lambda x.X(x) \times Y(x)$'. But this is the very same set:

(24)	$\downarrow[\lambda x.X(x) \times Y(x)]$	since $X(x) \times Y(x) = 1$ iff $X(x) = 1$ & $Y(x) = 1$
=	$\downarrow[\lambda x. \vdash X(x) = 1$ and $Y(x) = 1 \dashv]$	by (37) from 2.5
=	$\{x\mid X(x) = 1$ and $Y(x) = 1\}$	def. '$\downarrow$'
=	$\{x\mid x \in \downarrow X$ and $x \in \downarrow Y\}$	def. '$\cap$'
=	$\downarrow X \cap \downarrow Y$	

In order to express the right hand-side of (23) in type logic, a constant of type ***(et)t*** is used, which expresses that the extension of a predicate (given by the argument) characterizes a non-empty set. From predicate logic we import the symbol $\exists$ and the name *existential quantifier* for this constant.[9] Using common notation, we omit the lambda prefix and re-bracket whenever $\exists$ combines with a (type-logical) lambda term: if $\boldsymbol{\varphi}$ is a formula of type $\boldsymbol{t}$, instead of $\exists(\boldsymbol{\lambda x.\varphi})$ we write $(\exists \boldsymbol{x})\boldsymbol{\varphi}$, which reads: 'there is an x such that $\boldsymbol{\varphi}$ holds'.

Although the indirect interpretation of determiners will be addressed at length and in its proper context after we have familiarized ourselves with the

[8] Propositional logic, the beginnings of which go back to antiquity, is the area of formal logic that concerns the relations between conjunction and negation (which will be introduced in due course).

[9] Predicate logic (which goes back to Frege's 1879 *Begriffsschrift*) forms the core of formal logic; it covers propositional logic (mentioned in the previous footnote) but is less expressive and flexible than type logic.

formulae of type logic, we already offer a glimpse of the translation of the indefinite article:

(25) $(\boldsymbol{\lambda Q^{et}.(\lambda P^{et}.(\exists x^{e})[P(x) \wedge Q(x)])})$

Expanding (25) by undoing the abbreviations for conjunction and existential quantifier, it is readily shown (in an exercise) that (25) is a formula of type ***(et)((et)t)***.

The third logical constant is negation, which will be symbolised by $\neg$, as in propositional logic. Negation is of type $\boldsymbol{tt}$ and reverses truth values: if a formula $\boldsymbol{\varphi}$ of type $\boldsymbol{t}$ has truth value 1, $\boldsymbol{\neg(\varphi)}$ has truth value 0 – and vice versa. As in propositional logic, we usually omit the brackets around the argument of $\neg$ and simply write $\neg\boldsymbol{\varphi}$, which reads: 'it is not the case that $\boldsymbol{\varphi}$'. Given this convention, the type-logical translation of the determiner **kein-** runs as follows:

(26) $(\boldsymbol{\lambda Q^{et}.(\lambda P^{et}.\neg(\exists x^{e})[P(x) \wedge Q(x)])})$

The fourth and last logical constant will not receive a special motivation; its interpretation presents no difficulties and we will later recognize its value for indirect interpretation: identity between individuals, for which we will reserve a constant of type ***e(et)***, written by a boldface equality symbol. Hence $\boldsymbol{(\alpha = \beta)}$ stands for $\boldsymbol{= (\alpha)(\beta)}$, whenever $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are of type $\boldsymbol{e}$.

The construction rules (21) and the logical constants $\wedge$, $\exists$, $\neg$ and $=$ (including their types and the notational conventions) together make the syntactic side of type logic, i.e., the definition of the type-logical formulae. We now come to their interpretation.

5.3 The interpretation of type logic

Although we have seen how type-logical formulae are to be understood, they still lack a *systematic interpretation*. We will proceed as in the (direct) interpretation of natural language and assign to each formula $\boldsymbol{\alpha}$ a semantic value $\|\boldsymbol{\alpha}\|$ – $\boldsymbol{\alpha}$'s *denotation*. In the case of primitive formulae – constants and variables – this will be done directly, by way of *lexical equations* '$\|\boldsymbol{\alpha}\| = \ldots$' that specify the corresponding value; the value of complex formulae, on the contrary, systematically derives from the values of its (immediate) parts and the way they are combined – by application or abstraction.

There are two kinds of primitive formulae in type logic: variables and constants. As we will soon see, the interpretation of constants is a relatively simple affair. The variables, on the other hand, present a serious challenge to compositional interpretation. Roughly speaking, the problem is that a variable in isolation does not have a fixed extension. But in the context of a

complex formula, a variable may very well contribute to its interpretation. Thus a formula of the form $\boldsymbol{\lambda x}\ldots$ denotes a function that assigns its values to *arbitrary* objects of a certain type – but what exactly are *arbitrary* objects?

As one can see in (26), $\neg(\exists \boldsymbol{x}^{\boldsymbol{e}})[\boldsymbol{Q}(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})]$ can express that the sets characterised by $\boldsymbol{P}$ and $\boldsymbol{Q}$ are disjoint, but what does the $\boldsymbol{x}$ in this characterisation refer to? To arbitrary objects in the intersection of the sets mentioned – and thus to nothing (if they do happen to be disjoint)? Whatever mysteriously 'arbitrary' objects the $\boldsymbol{x}$ may refer to, they are dispensable for the (disjunction) statement made; for the underlined reformulation can do without them.

The problems with the function of variables and their compositional interpretation go way beyond type logic and cannot be solved without a certain amount of technical machinery. The solution we will adopt here is based on the idea that a variable simultaneously denotes all objects of its type.[10] Following this lead, a variable $\boldsymbol{x}$ of type $\boldsymbol{e}$ can refer to any individual. In other words, any individual may serve as the extension of $\boldsymbol{x}$; similarly, any predicate extension – i.e., any characteristic function of a set of individuals – can act as the extension of a variable $\boldsymbol{P}$ of type $\boldsymbol{et}$. Thus variables have more than one extension (in a given situation). The same is true of complex formulae containing variables:

(27) $\boldsymbol{P}(\boldsymbol{x})$

The formula in (27) can refer to any truth value that ensues if the – arbitrary – semantic value of $\boldsymbol{P}$ is applied to the – arbitrary – semantic value of $\boldsymbol{x}$.[11] So if (case 1) $\boldsymbol{P}$ denotes the predicate extension P_1 and $\boldsymbol{x}$ denotes the individual x_1, then (27) denotes $P_1(x_1)$; however, if (case 2) $\boldsymbol{P}$ and $\boldsymbol{x}$ respectively denote P_2 and x_2, then $P_2(x_2)$ is the semantic value of (27); but $\boldsymbol{P}$ may equally well (case 3) denote P_1 whereas $\boldsymbol{x}$ denotes x_2, whereupon the semantic value of (27) would of course be $P_1(x_2)$; etc. The cases on which it depends what the relevant variables stand for, are called *variable assignments* or merely *assignments* in logical semantics. Since an assignment says what each variable of each type stands for, one may think of it as a function that assigns semantic values to variables:

(28) *Definition*
An *assignment* is a function g whose arguments are the variables

[10] This method of interpreting variables, which is known as *assignment semantics* and originates with the Polish-American logician Alfred Tarski (1901–1983), is highly popular in logical semantics. An alternative – so called *substitution semantics* – will be sketched in Section 5.8.

[11] Our discussion presupposes that the construction *(App)* is interpreted as expressing functional application – which we will officially fix in a moment.

and such that the following holds: if $\boldsymbol{x}$ is a variable of some type a, then $g(\boldsymbol{x})$ is an object of type a.

Each of the cases 1–3 thus corresponds to a host of assignments: for instance, case 1 is given with any assignment g according to which $\boldsymbol{P}$ and $\boldsymbol{x}$ have the semantic values P_1 and x_1, i.e. as soon as $g(\boldsymbol{P}) = P_1$ and $g(\boldsymbol{x}) = x_1$ – no matter which values g assigns to the other variables.

As the discussion of (27) has shown, it is not enough to assume that only variables have several semantic values; the value of a complex expression can also depend on an assignment. The semantics of type logic thus does not merely have to assign a single semantic value to each expression $\boldsymbol{\alpha}$ but one semantic value $\|\boldsymbol{\alpha}\|^g$ for each assignment g. Thus if in case 2, where we have an assignment h with $h(\boldsymbol{P}) = P_2$ and $h(\boldsymbol{x}) = x_2$, the value of (27) comes out like this: $\|\boldsymbol{P(x)}\|^h = P_2(x_2)$. More generally, a given type-logical expression needs to obtain a separate semantic value for each assignment g. Unlike the interpretation of natural language given in the preceding chapters, the compositional interpretation of type logic will determine a whole 'family' of semantic values for each formula; and this family of values then makes the full meaning of the formula. In practice, though, only single values will play a rôle. Why this is so will be explained in due course.

Given these fundamental considerations on the architecture of the interpretation of type-logical formulae, the lexical equations for variables are now straightforward:

(29) *Interpretation of variables (Var)*
If g is an assignment and $\boldsymbol{x}$ is a variable, the following holds:
$\|\boldsymbol{x}\|^g = g(\boldsymbol{x})$

Since the type-logical translations of natural language expressions to be developed denote their extensions, they always contain a free variable referring to a possible situation. This will always be the same, fixed variable $\boldsymbol{i}$ of type $\boldsymbol{s}$; why this is so will become clear later. The assignment, which determines the value of the variable $\boldsymbol{i}$, thus also has the function of specifying the situation at hand.

Now for the constants. Having left open how many constants there are, and which ones, we cannot fix their interpretation either. Most constants will be introduced when required in indirect interpretation and will then receive their interpretation. There are, however, two constraints the lexical equations introducing them need to obey. For one thing, each constant has one (and only one) type, which fixes the type of its denotation; this constraint is analogous to the constraint imposed on variable assignments in (28). For another thing, the semantic value of a constant does, of course, not

depend on the semantic values of any variables; so given any two assignments, the constant will receive the same value relative to them. We record these constraints as the:

(30) *Constant Principles (Con)*

a. If g is an assignment and $\mathbf{c}$ is a constant of some type a, the following holds:
$\|\mathbf{c}\|^g$ is an object of type a.

b. If g and h are assignments and $\mathbf{c}$ is a constant (of any type), the following holds:
$\|\mathbf{c}\|^g = \|\mathbf{c}\|^h$.

All lexical equations must and will in the following satisfy the principles in (30). The following random example illustrates this:

(31) If g is an assignment, $\|\mathbf{f}\|^g$ = Fritz.

Under the (syntactic) assumption that $\mathbf{f}$ is a constant of type $\boldsymbol{e}$, (31) satisfies constraint (30a); after all, Fritz is an individual. Later equations will always satisfy (30a) without explicit mention. (30b) also holds; for (31) is understood to apply to all assignments g.

Given (30b), the superscript to the semantic brackets in the lexical equations for constants is redundant and will therefore be omitted in the future. Following this convention, (31) can be rewritten as:

(32) $\|\mathbf{f}\|$ = Fritz

Apart from this illustrative example, the only specific lexical equations for constants are those concerning the four logical constants; they run as follows:

(33) a. $\|\wedge\| = \lambda v.\lambda u.u \times v$
b. $\|\exists\| = \lambda P. \vdash \downarrow P \neq \emptyset \dashv$
c. $\|\neg\| = \lambda v.1 - v$
d. $\|=\| = \lambda x.\lambda y. \vdash x = y \dashv$

(33a)–(33d) make use of the convention of dropping the assignment superscripts; the reader is reminded that this is possible because we are dealing with constants, the interpretation of which is independent of any assignment.

The right hand-sides of the equations in (33a)–(33d) are all occupied by lambda-terms. These terms are no type-logical formulae but abbreviations for descriptions of functions, as we have used them in the previous chapters. The variables used in (33a)–(33d) are no type-logical variables either but variables of the meta-language that stand for arbitrary truth values ('u', 'v'), predicate extensions ('P'), and individuals ('x', 'y'), respectively. These

equations do not make explicit the types of objects that these meta-variables stand for; but they can be identified by constraint (30a), according to which, e.g., $\|\wedge\|$ must be a function of type $\boldsymbol{t(tt)}$ and thus (successively) take truth values as arguments; and since the variable 'v' following the lambda refers to the (first) argument of $\|\wedge\|$, it can only stand for truth values. The following, more cumbersome reformulations of the equations in (33) may be felt to be clearer in this respect:

(34) a. $\|\wedge\|$ = that function of type $\boldsymbol{t(tt)}$ that assigns to any truth value v that function of type $\boldsymbol{tt}$ that assigns to every truth value u the product $u \times v$;
b. $\|\exists\|$ = that function of type $\boldsymbol{(et)t}$ that assigns the truth value 1 to any predicate extension P just in case P does not characterise the empty set;
c. $\|\neg\|$ = that function of type $\boldsymbol{tt}$ that assigns to any truth value v the truth value $1 - v$;
d. $\|=\|$ = that function of type $\boldsymbol{e(et)}$ that assigns to any individual x that function of type $\boldsymbol{et}$ that assigns to any individual y the truth value 1 just in case x is identical to y.

Let us now turn to the interpretation of complex formulae. As already indicated, they are interpreted compositionally, which means that we will take the meanings of the immediate parts as given and specify the way in which they combine into the meaning of the whole formula. We need to distinguish two cases according to the construction rules for type-logical formulae. We start with the much simpler case in which two expressions $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are combined into $\boldsymbol{\alpha(\beta)}$. According to *(App)*, this is only possible if $\boldsymbol{\alpha}$ is a formula of a type ab, where $\boldsymbol{\beta}$ is of type a. We may thus assume that the semantic values $\|\boldsymbol{\alpha}\|^g$ and $\|\boldsymbol{\beta}\|^g$ are always – i.e., at any assignment g – objects of type ab and a, respectively. In particular, then, $\|\boldsymbol{\beta}\|^g$ is an argument to which $\|\boldsymbol{\alpha}\|^g$ assigns a value – and this value is what the entire formula denotes:

(35) *Interpretation of functional application (App)*
If g is an assignment and $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are, respectively, formulae of types ab and a, the following holds:
$\|\boldsymbol{\alpha(\beta)}\|^g = \|\boldsymbol{\alpha}\|^g(\|\boldsymbol{\beta}\|^g)$.

The second kind of complex formulae are the ones formed by *(Abs)*; they have the form $\boldsymbol{(\lambda x.\alpha)}$, where $\boldsymbol{x}$ is a variable of some type a and $\boldsymbol{\alpha}$ is a formula of some type b. In order to see how such formulae can be interpreted compositionally, let us first consider an example:

(36) $\boldsymbol{(\lambda y^e.(\lambda x^e.\mathbf{S}(i)(x)(y)))}$

Here **S** is a constant of type ***s(e(et))*** that denotes the intension of **sieht** [≈ sees]:

(37) $\|\mathbf{S}\| = \lambda s.\lambda y.\lambda x.\vdash x \text{ sees } y \text{ in } s \dashv$

Let us first note that the value of the formula (36) ought to be a function which, when applied to Eike, yields the predicate extension of **wird von Eike gesehen** [≈ is seen by Eike] (in the situation denoted by $\boldsymbol{i}$) – i.e., that function that assigns the truth value 1 to an individual x if in $g(\boldsymbol{i})$, x is seen by Eike; for if the extension of **sieht**, which is denoted by $\mathbf{S}(\boldsymbol{i})$, gets applied to x and then the result is applied to Eike, according to (37), the truth value 1 ensues just in case Eike sees x (in the situation denoted by $\boldsymbol{i}$). Thus (36) reverses, as it were, the relation of seeing denoted by $\mathbf{S}(\boldsymbol{i})$ into the relation of being seen by. In order to interpret (36) compositionally, the meaning of this formula must be obtained from the meanings of its two immediate parts – the variable $\boldsymbol{y}$ and the complex formula (38):

(38) $(\boldsymbol{\lambda x^e.\mathbf{S}(i)(x)(y)})$

The meaning of (38) in turn needs to be systematically obtained from the meaning of the variable $\boldsymbol{x}$ and the meaning of the formula (39):

(39) $\mathbf{S}(\boldsymbol{i})(\boldsymbol{x})(\boldsymbol{y})$

The denotation of (39) can be obtained successively by the above interpretation (35) of functional application.We thus get, for arbitrary assignments g:

(40)
$$\begin{aligned} & \|\mathbf{S}(\boldsymbol{i})(\boldsymbol{x})(\boldsymbol{y})\|^g \\ = \; & \|\mathbf{S}(\boldsymbol{i})(\boldsymbol{x})\|^g(\|\boldsymbol{y}\|^g) \\ = \; & \|\mathbf{S}(\boldsymbol{i})\|^g(\|\boldsymbol{x}\|^g)(\|\boldsymbol{y}\|^g) \\ = \; & \|\mathbf{S}\|^g(\|\boldsymbol{i}\|^g)(\|\boldsymbol{x}\|^g)(\|\boldsymbol{y}\|^g) \\ = \; & \|\mathbf{S}\|^g(g(\boldsymbol{i}))(g(\boldsymbol{x}))(g(\boldsymbol{y})) \end{aligned}$$

The last transition in (40) makes use of the interpretation (29) of variables according to which the denotation of a variable is determined by the assignmnt at hand. In (40), this dependence on the assignment is transmitted to the complex formula (39). Hence if g_1 assigns to the variables $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{i}$ the values Fritz, Eike, and s_0 – i.e., $g_1(\boldsymbol{x})$ = Fritz, $g_1(\boldsymbol{y})$ = Eike, and $g_1(\boldsymbol{i}) = s_0$ – the denotation of (39) at this assignment will be 1 if Eike sees Fritz in s_0; given an assignment g_2, according to which $g_2(\boldsymbol{i}) = g_1(\boldsymbol{i})$, and $g_2(\boldsymbol{x})$ = Fritz = $g_2(\boldsymbol{y})$, $\|(39)\|^{g_2} = 1$ if Fritz sees himself in s_0; etc.:

(41)

ass.	$\|\boldsymbol{i}\|^g$	$\|\boldsymbol{y}\|^g$	$\|\boldsymbol{x}\|^g$	$\|\mathbf{S}(\boldsymbol{i})(\boldsymbol{x})(\boldsymbol{y})\|^g$
g_1	s_0	Eike	Fritz	$\vdash$ Eike sees Fritz in s_0 $\dashv$
g_2	s_0	Fritz	Fritz	$\vdash$ Fritz sees himself in s_0 $\dashv$
g_3	s_0	Eike	Eike	$\vdash$ Eike sees herself in s_0 $\dashv$
g_4	s_1	Fritz	Eike	$\vdash$ Fritz sees Eike in s_1 $\dashv$
g_5	s_1	Fritz	Fritz	$\vdash$ Fritz sees himself in s_1 $\dashv$
...	...	...	...	...
g	$g(\boldsymbol{i})$	$g(\boldsymbol{y})$	$g(\boldsymbol{x})$	$\vdash$ $g(\boldsymbol{y})$ sees $g(\boldsymbol{x})$ in $g(\boldsymbol{i})$ $\dashv$
...	...	...	...	...

In passing from (39) to (38) this assignment dependence is reduced by the lambda-operator. For (38) refers to a function that assigns the truth value 1 to *arbitrary* individuals x if x sees the individual denoted by the variable $\boldsymbol{y}$ in the situation denoted by the variable $\boldsymbol{i}$:

(42)

ass.	$\|\boldsymbol{i}\|^g$	$\|\boldsymbol{y}\|^g$	$\|\boldsymbol{x}\|^g$	$\|\boldsymbol{\lambda x}.\mathbf{S}(\boldsymbol{i})(\boldsymbol{x})(\boldsymbol{y})\|^g$
g_1	s_0	Eike	Fritz	$\lambda x.\vdash$ Eike sees x in s_0 $\dashv$
g_2	s_0	Fritz	Fritz	$\lambda x.\vdash$ Fritz sees x in s_0 $\dashv$
g_3	s_0	Eike	Eike	$\lambda x.\vdash$ Eike sees x in s_0 $\dashv$
g_4	s_1	Fritz	Eike	$\lambda x.\vdash$ Fritz sees x in s_1 $\dashv$
g_5	s_1	Fritz	Fritz	$\lambda x.\vdash$ Fritz sees x in s_1 $\dashv$
...	...	...	...	...
g	$g(\boldsymbol{i})$	$g(\boldsymbol{y})$	$g(\boldsymbol{x})$	$\lambda x.\vdash$ $g(\boldsymbol{y})$ sees x in $g(\boldsymbol{i})$ $\dashv$
...	...	...	...	...

So while the value of (39) depends on who or what the variables $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{i}$ refer to, only the denotations of $\boldsymbol{y}$ and $\boldsymbol{i}$ matter for (38): as soon as two assignments agree on these variable-values (which is indicated by a match in colouring), they also agree on the denotation of the entire formula. In comparison to the interpretation of (39) shown in (41), the lambda in (38) thus 'neutralizes' the denotation of $\boldsymbol{x}$. In formal logic, this form of neutralization of truth values is called *binding*: the initially *free* variable in the *scope* of the lambda-operator gets *bound* by it in abstraction. Here, the scope (aka the *matrix*) is that part of the formula that the operator operates on – between full stop and bracket. It should be noted that the status of a variable – whether it is free or bound – is something relative; a variable is never free or bound 'as such' but only in relation to a formula: thus, e.g., $\boldsymbol{x}$ is free in (39) but not in (38).

Binding is a very general process that occurs whenever variables are used to seemingly refer to arbitrary objects. In order to see how this process works, let us take a closer look at the transition from (41) to (42), first concentrating on g_1. Given this assignment, the denotation that the formula (38) has according to (42), is a function that can again be represented by a

table:

(43) $\|(\boldsymbol{\lambda x.S(i)(x)(y)})\|^{g_1} =$

argument	*value*
Fritz	⊢ Eike sees Fritz in s_0 ⊣
Eike	⊢ Eike sees herself in s_0 ⊣
...	...
NN	⊢ Eike sees *NN* in s_0 ⊣

(43) represents the denotation in the last line of (42). The overall goal of our current investigation is to find a general meaning combination that produces the denotations of formulae of the form $\boldsymbol{(\lambda z.\alpha)}$ from those of the bound variables $\boldsymbol{z}$ and the matrix formula $\boldsymbol{\alpha}$. So we should find out how the function displayed in (43) can be obtained from the meaning of the matrix $\mathbf{S}(\boldsymbol{i})(\boldsymbol{x})(\boldsymbol{y})$ represented in (41). Actually, this is not so hard if we carry over the shading from (42) to (41): (43) emerges from the two rightmost columns of (41) by eliminating those lines that differ in their shading from the first (g_1-) line (where):

(44)

ass.	$\|\boldsymbol{i}\|^g$	$\|\boldsymbol{y}\|^g$	$\|\boldsymbol{x}\|^g$	$\|\mathbf{S}(\boldsymbol{i})(\boldsymbol{x})(\boldsymbol{y})\|^g$
g_1	s_0	Eike	Fritz	⊢ Eike sieht Fritz in s_0 ⊣
~~g_2~~	~~s_0~~	~~Fritz~~	~~Fritz~~	~~⊢ Fritz sieht sich in s_0 ⊣~~
g_3	s_0	Eike	Eike	⊢ Eike sieht sich in s_0 ⊣
~~g_4~~	~~s_1~~	~~Fritz~~	~~Eike~~	~~⊢ Fritz sieht Eike in s_1 ⊣~~
~~g_5~~	~~s_1~~	~~Fritz~~	~~Fritz~~	~~⊢ Fritz sieht sich in s_1 ⊣~~
...	...	...	...	...

The same procedure also leads from (41) to the denotation of $\boldsymbol{(\lambda x^e.S(i)(x)(y))}$ given g_3, since this assignment has the same colour; once all differently shaded lines have been eliminated, (43) emerges again – and indeed, according to (42), $\|(\boldsymbol{\lambda x.S(i)(x)(y)})\|^{g_1} = \|(\boldsymbol{\lambda x.S(i)(x)(y)})\|^{g_3}$. In a similar vein, the denotation of $\boldsymbol{(\lambda x^e.S(i)(x)(y))}$ given the assignments g_4 or g_5 emerges if one only keeps the darkened assignments in (41) and then cuts out the mapping represented by the two rightmost columns. Writing up the tables has been delegated to an exercise. In the end, the following general pattern emerges: the denotation of a formula of the form $\boldsymbol{(\lambda z.\alpha)}$ given an assignment g, is a function f, that assigns to the values of $\boldsymbol{z}$ the denotation of $\boldsymbol{\alpha}$ at the assignments matching with g in shading. Hence if h is such a matching assignment, f assigns to an object $u = h(\boldsymbol{z})$ the denotation of $\boldsymbol{\alpha}$ given h:

(45) $\|(\boldsymbol{\lambda z.\alpha})\|^g(h(\boldsymbol{z})) = \|\boldsymbol{\alpha}\|^h$

(45) basically accounts for the meaning combination expressed by lambda abstraction that we are after. However, the equation needs to be understood as applying to *arbitrary* assignments g and h *with the same shading*. But

what does a match in shading amount to in general? In the above tables, two assignments always received the same shading if they agree on the values for $\boldsymbol{i}$ and $\boldsymbol{y}$ – or rather (and more generally speaking): *if they differ at most in their* $\boldsymbol{x}$*-value*. The function denoted by the lambda-term could be constructed from the assignments with the same shading precisely because each individual appears as an $\boldsymbol{x}$-value among them, without thereby affecting the values of any of the other variables. Using standard terminology from formal logic, we call two assignments that differ at most in the values of one variable $\boldsymbol{x}$, $\boldsymbol{x}$*-alternatives* of each other. Given this term, we may generalize (45) to the following meaning rule for lambda-formulae:

(46) *Interpretation of abstraction (Abs)* [initial version]
If g is an assignment, $\boldsymbol{x}$ is a variable of type a, and $\boldsymbol{\alpha}$ is a formula of type b, then $\|\boldsymbol{(\lambda x.\alpha)}\|^g$ is that function of type (ab) such that for any $\boldsymbol{x}$-alternative h of g the following holds:
$\|\boldsymbol{(\lambda x.\alpha)}\|^g(h(\boldsymbol{x})) = \|\boldsymbol{\alpha}\|^h$

According to (46), the denotation of the lambda-term does come out as a function that assigns to any object of type a an object of type b. For any object u of type a is the value of an $\boldsymbol{x}$-alternative to g [for the argument $\boldsymbol{x}$], viz. *of that function that is like* g *except that it assigns to the variable* $\boldsymbol{x}$ *the object* u. It is readily seen there is *exactly one* $\boldsymbol{x}$-alternative that does that.[12] We will write it as '$g[\boldsymbol{x}/u]$' and call it a *modification of* g. Using this notation, (46) can be reformulated thusly:

(47) *Interpretation of abstraction (Abs)* [standard version]
If g is an assignment, $\boldsymbol{x}$ is a variable of type a, and $\boldsymbol{\alpha}$ is a formula of type b, then $\|\boldsymbol{(\lambda x.\alpha)}\|^g$ is that function of type (ab) such that for any object u of type a the following holds:
$\|\boldsymbol{(\lambda x.\alpha)}\|^g(u) = \|\boldsymbol{\alpha}\|^{g[\boldsymbol{x}/u]}$

The reader should realise that (47) and (46) do say the same thing; but the reformulation makes it clearer that $\|\boldsymbol{(\lambda x.\alpha)}\|^g$ is indeed an object of type (ab), i.e., a function from the objects of type a to the objects of type b. This is why (47) is a common textbook formulation.

Under the assumption that the meta-linguistic variable 'u' ranges over all objects of a given type a, (47) can also be reformulated with the help of the meta-linguistic lambda-operator:

(48) *Interpretation of abstraction (Abs)* [quick version]
If g is an assignment, $\boldsymbol{x}$ is a variable of type a, and $\boldsymbol{\alpha}$ is a formula of type b, then:

[12] We are assuming that h and h' are the same function if they agree on all variables. In set theory, this justified by the Principle of Extensionality (7) of Chapter 1.

$$\|(\boldsymbol{\lambda x.\alpha})\|^g = \lambda u.\|\boldsymbol{\alpha}\|^{g[\boldsymbol{x}/u]}$$

(48) brings out clearly that the $\boldsymbol{\lambda}$-Operator of type logic corresponds to the lambda-notation used in the meta-language.

The ynterpretation mechanism given in (46)–(48) can also be used to determine the denotation of the initial formula (36), repeated below, from the denotations of the subformula $(\boldsymbol{\lambda x.S(i)(x)(y)})$ given in Table (42) above:

(36) $\quad (\boldsymbol{\lambda y^e.(\lambda x^e.S(i)(x)(y))})$

After finally binding the situation variable $\boldsymbol{i}$, it turns out (in an exercise) that the denotation of the resulting formula will be the same, whatever the assignment may be. Small wonder: the formula contains no free variables – in logical terms, it is *closed*.

Are (46)–(48) compositional? Yes and no. On the one hand, according to these rules, the denotation of a lambda-formula (given an assignment g) is not composed from the denotations of their immediate parts. Rather, to determine it, the denotations of the parts at other assignments – the $\boldsymbol{x}$-alternatives or modifications – need to be called in. On the other hand, according to (46)–(48), the *totality of all denotations* of a lambda-formula at all assignments can be determined from the *totality of all denotations* of its parts – and these totalities, which can themselves be construed as functions that assign denotations to assignments, may be taken as the meanings of type-logical formulae. More precisely, the *meaning* of a type-logical formula $\boldsymbol{\alpha}$ may be identified with the function that maps any assignment g to the denotation $\boldsymbol{\alpha}$ has (given g) – set-theoretically speaking, the set of pairs $(g, \|\boldsymbol{\alpha}\|^g)$. Though we will not make explicit reference to this notion of meaning, we will be relying on its compositionality throughout the remainder of these notes.

It is not the denotations of the sub-formulae that get combined by variable-binding but their entire meanings. However, (46) and (47) only indicate how the meaning of the right part – the matrix $\boldsymbol{\alpha}$ – enters the compositional determination of the denotation of the lambda-term. But the above (rather common) formulations remain silent about the rôle the alternative values of $\boldsymbol{x}$ play. At least they are not mentioned explicitly in (47); and though they do enter the main equation (45) in (46), it does not really become clear how the denotation of the entire formula is determined *only* by them and the values of the matrix. In order to fully settle the compositionality issue about variable binding, (46) and (47) would thus have to be reformulated once more. To this end, the notion of an '$\boldsymbol{x}$-alternative' or 'being an assignment that is modified at $\boldsymbol{x}$' needs to be replaced by concepts that only relate to the meaning of the variable $\boldsymbol{x}$, and not to the variable itself. That such a formulation is possible

will be shown in an exercise.[13]

5.4 Logical transformations

As already mentioned at the beginning of the chapter, one huge advantage of indirect interpretation is its transparency. The more complex the meanings to be determined and manipulated get, the more desirable – and even necessary – it will be to have a transparent representation for them. To a large extent, the transparency of the method is owed to the fact that type-logical formulae respect certain logical laws that allow for quasi-mechanical reductions of complex formulae, thus dramatically increasing their readability. These reductions are what the current section is about. The logical laws behind them are by no means additional interpretive constraints, let alone arbitrary notational conventions. On the contrary, they are direct consequences of the interpretation of the formulae given in the preceding section, which implies in a host of cases that two given formulae are *logically equivalent* in that they always – i.e. given any assignment – denote the same object. Here is a case in point:

(49) If $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ are formulae of type $\boldsymbol{t}$, then:
$[\boldsymbol{\varphi} \wedge \boldsymbol{\psi}] \equiv [\boldsymbol{\psi} \wedge \boldsymbol{\varphi}]$

In (49), the symbol '$\equiv$' stands for said relation of logical equivalence. It should be noted that it is not a type-logical symbol but a meta-linguistic abbreviation. In particular, the line on display in (49) is not itself a type-logical formula but abbreviates the following meta-linguistic statement:

(50) For all assignments g the following holds: $\|[\boldsymbol{\varphi} \wedge \boldsymbol{\psi}]\|^g = \|[\boldsymbol{\psi} \wedge \boldsymbol{\varphi}]\|^g$.

There is a close connection between logical equivalence and the notion of meaning defined at the end of the preceding section:[14] as it turns out, two type-logical expressions are logically equivalent just in case they have the same meaning. In other words, synonymy, understood as sameness of meaning, boils down to logical equivalence – having the same denotation given any assignment. This connection will become important in the process of indirect interpretation, and particularly when it comes to the reduction of type-logical translations in the interest of readibility. For, as indicated above, meanings behave compositionally. As a consequence, a formula can be replaced by a logically equivalent one in any (larger) formula without affecting

[13] A more detailed account of the Tarskian approach to compositionality based on meanings as functions on assignments can be found in the tenth chapter of the *Introduction to Semantics* by T. E. Zimmermann and W. Sternefeld (Berlin 2013).

[14] This calls for some qualification, given that our use of these terms diverts from the slightly narrower ('model-theoretic') definitions that they receive in formal logic.

the meaning of the latter. We will take advantage of this fact by applying logical transformations not only directly to entire translations but also to their parts.

In what sense does (49) follow from the above interpretation of type-logical formulae? The answer is simple: the claim can be *proved*; somewhat pompously put, it is a mathematical *theorem*. In order to convey an impression of how such transformation laws like (49) may in principle be justified, and that in doing so, one relies solely on the compositional interpretation of the formulae, we will give an exemplary demonstration of (49); afterwards we will do without such detailed proofs.

Let us thus suppose we are given some formulae $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ of type $\boldsymbol{t}$. In order to show the equivalence claimed in (49), we need to consider an arbitrary assignment g and compare the denotations of $[\boldsymbol{\varphi} \wedge \boldsymbol{\psi}]$ and $[\boldsymbol{\psi} \wedge \boldsymbol{\varphi}]$ given g – i.e. $\|[\boldsymbol{\varphi} \wedge \boldsymbol{\psi}]\|^g$ and $\|[\boldsymbol{\psi} \wedge \boldsymbol{\varphi}]\|^g$. As a first step, we should undo the notational simplifications; '$[\boldsymbol{\varphi} \wedge \boldsymbol{\psi}]$' stands for the type-logical formula $\wedge\boldsymbol{(\psi)(\varphi)}$, starting wiht the logical constant $\wedge$ of type $\boldsymbol{t(tt)}$, the meaning of which was given in (33a), repeated here for the readers' convenience:

(33a) $\quad \|\wedge\| = \lambda v.\lambda u.u \times v$

We now note that $\wedge\boldsymbol{(\psi)(\varphi)}$ is the result of two applications of the syntactic rule *(App)*: first, the formula $\wedge\boldsymbol{(\psi)}$ of type $\boldsymbol{tt}$ is formed from the constant $\wedge$ – which, following *(Con)*, is a formula of type $\boldsymbol{t(tt)}$ – and the formula $\boldsymbol{\psi}$; then the final result ensues from the latter and the formula $\boldsymbol{\varphi}$. (If you find following this reasoning difficult, you should draw a syntactic tree!) And, to be sure, in both cases the construction rule *(App)* is the source. This being so, the denotation of $\wedge\boldsymbol{(\psi)(\varphi)}$ can be determined stepwise, using the interpretation (35) of formulae built by *(App)*, as introduced in the preceding section and repeated here:

(35) *Interpretation of functional application (App)*
If g is an assignment and $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are, respectively, formulae of types ab and a, the following holds:
$\|\boldsymbol{\alpha(\beta)}\|^g = \|\boldsymbol{\alpha}\|^g(\|\boldsymbol{\beta}\|^g)$.

Here is how it goes:[15]

[15]Our proof contains two applications of λ-conversion. In the interest of clarity, it should be pointed out that this is not necessary. Instead one may recall that meta-linguistic lambda-terms abbreviate descriptions of functions, and then insert these descriptions themselves. Thus, e.g., the λ-term '$[\lambda u.u \times \|\boldsymbol{\psi}\|^g]$' abbreviates 'that function f that assigns to any truth value u the value $u \times \|\boldsymbol{\psi}\|^g$'; and '$[\lambda u.u \times \|\boldsymbol{\psi}\|^g](\|\boldsymbol{\varphi}\|^g)$' denotes the value of the function thus described for the specific argument $u = \|\boldsymbol{\varphi}\|^g$ – i.e. $\|\boldsymbol{\varphi}\|^g \times \|\boldsymbol{\psi}\|^g$.

(51) $\|\wedge(\boldsymbol{\psi})(\boldsymbol{\varphi})\|^g$

$= \|\wedge(\boldsymbol{\psi})\|^g(\|\boldsymbol{\varphi}\|^g)$ by *(App)*: $\wedge(\boldsymbol{\psi})$ for $\boldsymbol{\alpha}$, $\boldsymbol{\varphi}$ for $\boldsymbol{\beta}$

$= \|\wedge\|^g(\|\boldsymbol{\psi}\|^g)(\|\boldsymbol{\varphi}\|^g)$ by *(App)*: $\wedge$ for $\boldsymbol{\alpha}$, $\boldsymbol{\psi}$ for $\boldsymbol{\beta}$

$= [\lambda v.\lambda u.u \times v](\|\boldsymbol{\psi}\|^g)(\|\boldsymbol{\varphi}\|^g)$ by (33a)

$= [\lambda u.u \times \|\boldsymbol{\psi}\|^g](\|\boldsymbol{\varphi}\|^g)$ λ-conversion

$= \|\boldsymbol{\varphi}\|^g \times \|\boldsymbol{\psi}\|^g$ λ-conversion

Since the equation (51) holds for arbitrary formulae $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ of type $\boldsymbol{t}$, it also holds for arbitrary formulae $\boldsymbol{\psi}$ und $\boldsymbol{\varphi}$ of this type:

(52) $\|\wedge(\boldsymbol{\varphi})(\boldsymbol{\psi})\|^g = \|\boldsymbol{\psi}\|^g \times \|\boldsymbol{\varphi}\|^g$

But then this is the very same result as the one calculated in (51) – viz. the (arithmetical) product of the truth values $\|\boldsymbol{\varphi}\|^g$ and $\|\boldsymbol{\psi}\|^g$. This concludes the proof of the equivalence claimed in (49) – the so-called *commutativity* of conjunction.

The transformation (49) is totally trivial; it is immediate without calling for a laborious proof. And it does not really help to achieve simplicity: the two formulae it equates have the same length, after all. But then the purpose of the above explicit equivalence proof was not the result reached but the path leading to it: it shows that in principle, the logical equivalence can be proved on the basis of the definitions given in the previous section.

Many of the most important rules of logical transformation refer to the difference between free and bound variables. It is a prime characteristic of variable binding that it makes the value of the variable independent of the assignment. This observation is the content of a fundamental lemma:[16]

(53) *Coincidence Lemma*
Let $\boldsymbol{\alpha}$ be a type-logical formula. Then for all assignments g and h that agree on the free variables of $\boldsymbol{\alpha}$ the following holds:
$\|\boldsymbol{\alpha}\|^g = \|\boldsymbol{\alpha}\|^h$.

That two assignments g and h *agree* on a variable $\boldsymbol{x}$ obviously means that $g(\boldsymbol{x}) = h(\boldsymbol{x})$. Hence the Coincidence Lemma says that the agreement of the free variables carries over to the whole formula. Plainly, this means that the denotation of the formula $\boldsymbol{\alpha}$ only depends on the variables free in $\boldsymbol{\alpha}$ – and thus neither on the variables that do not occur in $\boldsymbol{\alpha}$ at all, nor on those that only occur as bound variables (in $\boldsymbol{\alpha}$). And this is precisely the above observation about neutralizing assignment dependence by binding.

(53) can be proved stepwise, by first establishing it for primitive, non-

[16] In the literature, the Coincidence *Lemma* is also referred to as Coincidence *Theorem*. The 'model-theoretic' version found in logic texts (cf. footnote 14) is slightly more general.

compound formulae, and then showing how complex formulae inherit the assignment independence claimed in the Coincidence Lemma from their immediate parts. The strategy is thus the principle of *mathematical induction* familiar from arithmetic.[17] We skip the proof and only notice that it can only be given on the basis of a precise definition of the concept of a *free variable*. This concept, too, can be defined stepwise, by specifying for each formula $\boldsymbol{\alpha}$ a set $Fr(\boldsymbol{\alpha})$:

(54) *Definition of the free variables of a formula* $\boldsymbol{\alpha}$

formula $\boldsymbol{\alpha}$	*formation rule*	$Fr(\boldsymbol{\alpha})$
$\boldsymbol{c}$	*(Con)*	$\emptyset$
$\boldsymbol{x}$	*(Var)*	$\{\boldsymbol{x}\}$
$\boldsymbol{\beta(\gamma)}$	*(App)*	$Fr(\boldsymbol{\beta}) \cup Fr(\boldsymbol{\gamma})$
$\boldsymbol{(\lambda x.\beta)}$	*(Abs)*	$Fr(\boldsymbol{\beta})\backslash\{\boldsymbol{x}\}$

It can be gleaned from Table (54) which variables occur freely in a given type-logical formula $\boldsymbol{\alpha}$. If $\boldsymbol{\alpha}$ is a constant (line 1), the set of the variables occurring freely in $\boldsymbol{\alpha}$ is empty – for no variables occur as part of a constant. If however, $\boldsymbol{\alpha}$ is itself a variable (line 2), it obviously occurs in $\boldsymbol{\alpha}$ – and freely so, for there are no $\boldsymbol{\lambda}$s in $\boldsymbol{\alpha}$ that could bind it. A variable occurring freely in a functional application $\boldsymbol{\beta(\gamma)}$ needs to occur in (at least) one of its parts; the set of the variables occurring freely in $\boldsymbol{\beta(\gamma)}$ thus results from taking the union of $Fr(\boldsymbol{\beta})$ and $Fr(\boldsymbol{\gamma})$. And the free variables of an abstraction $\boldsymbol{(\lambda x.\beta)}$ are those of its matrix $\boldsymbol{\beta}$ – with the obvious exception of the $\boldsymbol{\lambda}$-bound $\boldsymbol{x}$.

That a variable is free in a formula $\boldsymbol{\alpha}$ does not mean that it does also occur as a bound variable in $\boldsymbol{\alpha}$ not at the same time; the variable may, after all, occur in several positions – as, e.g., the $\boldsymbol{p}$ in $\boldsymbol{(\lambda p^t.[p \wedge q^t])(p)}$. A formula that does not contain any free variables is called *closed*; accordingly, *open* formulae are those with free variables. The Coincidence Lemma immediately implies that the denotations of closed formulae are always assignment-independent; for if g and h are any assignments and $\boldsymbol{\alpha}$ is a closed formula, then g and h trivially agree on all free variables of $\boldsymbol{\alpha}$, given that $Fr(\boldsymbol{\alpha})$ is empty. In the case of closed formulae $\boldsymbol{\alpha}$, the denotation can be simply written as '$\|\boldsymbol{\alpha}\|$', as we already did for constants (which are, of course, special cases of closed formulae). But then we will rarely have the opportunity to make use of this notational convention: as it will turn out, the type-logical translations of natural language expression typically contain the free situation variable $\boldsymbol{i}$. However, as a rule, $\boldsymbol{i}$ will be the only variable that occurs freely in a type-logical translation $\boldsymbol{\alpha}$, i.e. $Fr(\boldsymbol{\alpha}) \subseteq \{\boldsymbol{i}\}$. The assignment thus only plays a rôle for such $\boldsymbol{\alpha}$ in assigning a denotation to the variable $\boldsymbol{i}$; for according to the Coincidence Lemma $\|\boldsymbol{\alpha}\|^g = \|\boldsymbol{\alpha}\|^h$ whenever $g(\boldsymbol{i}) = h(\boldsymbol{i})$. So in specifying

[17] This is the principle according to which every number has a given property P as soon as P holds of the number 0 and carries over from any number (n) to the next one $(n+1)$.

the denotation of such $\boldsymbol{\alpha}$, we need not consider the whole assignment but only its value for $\boldsymbol{i}$. In the future we will thus sometimes write '$\|\boldsymbol{\alpha}\|^s$' for the denotation of a formula $\boldsymbol{\alpha}$, given an (arbitrary) assignment g for which: $g(\boldsymbol{i}) = s \in LS$. It should be noted that this new notation is reminiscent of the extensions of natural-language expressions and that it only makes sense for type-logical formulae $\boldsymbol{\alpha}$ for which $Fr(\boldsymbol{\alpha}) \subseteq \{\boldsymbol{i}\}$.

Variable binding creates assignment independence. This is one of two essential features of this process. The second one is equally familiar from the pre-theoretic treatment of variables, viz. that their precise identity is immaterial: it does not matter whether we are talking of an arbitrary x or an arbitrary y – bound variables can be more or less randomly renamed without affecting the content of the overall statement. This insight will be recorded by way of a logical transformation which is based on a principle that is as fundamental for the understanding of variable binding as the Coincidence Lemma. The principle concerns the substitution of *free* variables by arbitrary formulae of the same type, and says that the result of such a substitution does not affect the denotation of the overall formula (in which the substitution took place) as long as the variable inserted has the same value as the formula that it replaces. This may sound complicated, but it isn't really. Let us consider an example:

(55) $\mathbf{T}(\boldsymbol{i})(\boldsymbol{x}^{\boldsymbol{e}})(\mathbf{e})$

In (55), $\mathbf{T}$ is the same constant of type $\boldsymbol{s(e(et))}$ that features in the translation of **trifft** [≈ is meeting], and $\mathbf{e}$ is a constant of type $\boldsymbol{e}$ whose denotation is Eike (at any assignment). So (55) is an open formula of type $\boldsymbol{t}$ – $\boldsymbol{x}$ and $\boldsymbol{i}$ are free in it – and its denotation given an assignment g obviously depends on which individual g assigns to the variable $\boldsymbol{x}$ and what the relevant situation $g(\boldsymbol{i})$ is: if $g(\boldsymbol{x}) =$ Eike and $g(\boldsymbol{i}) = s^*$, then $\|\mathbf{T}(\boldsymbol{i})(\boldsymbol{x})(\mathbf{e})\|^g$ is the truth value 1 if Eike is meeting herself in s^*;[18] if $h(\boldsymbol{x}) =$ Fritz and $h(\boldsymbol{i}) = s^+$, then $\|\mathbf{T}(\boldsymbol{i})(\boldsymbol{x})(\mathbf{e})\|^h$ is the value 1 if Eike meets Fritz in s^+; etc. Now, if $\mathbf{f}$ is a constant of type $\boldsymbol{e}$ denoting Fritz, it is clear that, given the assignment h at hand, the formula (55) has the same denotation as:

(56) $\mathbf{T}(\boldsymbol{i})(\mathbf{f})(\mathbf{e})$

(56) derives from (55) by substituting the (free) variable $\boldsymbol{x}$ by the constant $\mathbf{f}$. Since the two have the same denotation given the assignment h, viz. Fritz, this denotational equivalence carries over to the entire formula. The general principle behind this is called the *Substitution Lemma*. In order to

[18] This has to be understood in the (literal) reading of **meet**, which more or less coincides with German **treffen** as well as German **begegnen**: it appears that the only way one can meet oneself in this sense is by way of time traveling (– or is it?). On a different reading, German **treffen** translates as **hit** and would thus allow for less absurd situations s^*, but some rather macabre ones.

formulate it, one needs a general concept of substitution of free variables: if $\boldsymbol{\alpha}$ is any formula, '$\boldsymbol{\alpha}[x/\delta]$' refers to the formula that ensues if all free occurrences of the variable $\boldsymbol{x}$ are replaced by the formula $\boldsymbol{\delta}$ (of the same type). Given this notation, the transition from (55) to (56) can be described as: $\mathbf{T(}\boldsymbol{i}\mathbf{)(}\boldsymbol{x}^{\boldsymbol{e}}\mathbf{)(e)}[x/\mathrm{f}] = \mathbf{T(}\boldsymbol{i}\mathbf{)(f)(e)}$. Substitution, too, can be defined stepwise, i.e., inductively:

(57) *Definition of substituting free* $\boldsymbol{x}$ *in a formula* $\boldsymbol{\alpha}$ *by* $\boldsymbol{\delta}$

	formula $\boldsymbol{\alpha}$	*formation rule*	$\boldsymbol{\alpha}[x/\delta]$
1.	$\mathbf{c}$	*(Con)*	$\mathbf{c}$
2.	$\boldsymbol{x}$	*(Var)*	$\boldsymbol{\delta}$
	$\boldsymbol{y}$ $[\boldsymbol{y} \neq \boldsymbol{x}]$	*(Var)*	$\boldsymbol{y}$
3.	$\boldsymbol{\beta(\gamma)}$	*(App)*	$\boldsymbol{\beta}[x/\delta](\boldsymbol{\gamma}[x/\delta])$
4.	$\boldsymbol{(\lambda x.\beta)}$	*(Abs)*	$\boldsymbol{(\lambda x.\beta)}$
	$\boldsymbol{(\lambda y.\beta)}$ $[\boldsymbol{y} \neq \boldsymbol{x}]$	*(Abs)*	$\boldsymbol{(\lambda y.\beta}[x/\delta]\boldsymbol{)}$

Table (57) shows what happens if all free occurrences of a variable $\boldsymbol{x}$ in a given type-logical formula $\boldsymbol{\alpha}$ are replaced by a formula $\boldsymbol{\delta}$. To make sure that the result $\boldsymbol{\alpha}[x/\delta]$ of this substitution is a type-logical formula at all, it is assumed that $\boldsymbol{x}$ and $\boldsymbol{\delta}$ are formulae of the same type; however, $\boldsymbol{\alpha}$ may be of a different type, as illustrated by (56), where free $\boldsymbol{x}$ of type $\boldsymbol{e}$ has been replaced by the constant $\mathbf{f}$ (of the same type) within the entire formula (55) of type $\boldsymbol{t}$. If $\boldsymbol{\alpha}$ is a constant (line 1), nothing happens when free $\boldsymbol{x}$ in $\boldsymbol{\alpha}$ is substituted – for neither $\boldsymbol{x}$ nor any other variable occurs in $\boldsymbol{\alpha}$; consequently, $\boldsymbol{\alpha}$ itself is the result of the substitution. If, however, $\boldsymbol{\alpha}$ happens to be the variable $\boldsymbol{x}$ (line 2a), the latter does occur in $\boldsymbol{\alpha}$ once, and replacing this one occurrence by $\boldsymbol{\delta}$ results in $\boldsymbol{\delta}$. If, on the other hand, $\boldsymbol{\alpha}$ is a different variable (line 2b), distinct from $\boldsymbol{x}$, we are in a similar situation as in the first case (line 1) – and nothing happens. If one wants to replace the variable $\boldsymbol{x}$ in an application $\boldsymbol{\beta(\gamma)}$ (line 3) by $\boldsymbol{\delta}$ wherever it occurs freely, one has to do so in both immediate parts of the formula – the functor $\boldsymbol{\beta}$ and the argument term $\boldsymbol{\gamma}$ – and then combine the results of these substitutions by functional application again. Abstractions $\boldsymbol{(\lambda y.\beta)}$ in which the bound variable does not happen to be $\boldsymbol{x}$ (line 4b), are treated in a similar way: the substitution needs to be performed in the matrix $\boldsymbol{\beta}$ and then the variable (distinct from $\boldsymbol{x}$) gets abstracted again. If however, $\boldsymbol{x}$ itself is bound by $\boldsymbol{\lambda}$ (line 4a), nothing happens; for the $\boldsymbol{\lambda}$ leaves no free occurrences of $\boldsymbol{x}$ to be substituted.

The reader should verify that according to the definition in Table (57), we do have: $\mathbf{T(}\boldsymbol{i}\mathbf{)(}\boldsymbol{x}^{\boldsymbol{e}}\mathbf{)(e)}[x/\mathrm{f}] = \mathbf{T(}\boldsymbol{i}\mathbf{)(f)(e)}$. So if the variable to be substituted has the same denotation as the constant replacing it (given an assignment), this denotational equivalence ought to carry over to the entire formula (before and after substitution). However, not always does the substitution of (free) variables by co-designating formulae lead to an equivalent result; in other

words, the following principle is not generally valid:

(58) If $\|\boldsymbol{x}\|^g = \|\boldsymbol{\delta}\|^g$, then $\|\boldsymbol{\alpha}\|^g = \|\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{\delta}]\|^g$.

Here is a simple counter-example. Choose $\boldsymbol{\alpha}$ to be the formula $(\boldsymbol{\lambda y^e.\mathbf{T}(i)(y)(x^e)})$ and consider an assignment g according to which $g(\boldsymbol{x}) = g(\boldsymbol{y})$ = Eike. Then $\|\boldsymbol{x}\|^g = \|\boldsymbol{y}\|^g$, and with (58) – putting $\boldsymbol{\delta} = \boldsymbol{y}$ – we would get:

(59) $\|(\boldsymbol{\lambda y^e.\mathbf{T}(i)(y)(x^e)})\|^g = \|(\boldsymbol{\lambda y^e.\mathbf{T}(i)(y)(x^e)})[\boldsymbol{x}/\boldsymbol{y}]\|^g$.

But $(\boldsymbol{\lambda y.\mathbf{T}(i)(y)(x)})[\boldsymbol{x}/\boldsymbol{y}] = (\boldsymbol{\lambda y.\mathbf{T}(i)(y)(y)})$ – which need not have the same denotation as $(\boldsymbol{\lambda y.\mathbf{T}(i)(y)(x)})$, despite the common assignment value of $\boldsymbol{x}$ and $\boldsymbol{y}$: applying $\|(\boldsymbol{\lambda y.\mathbf{T}(i)(y)(x)})\|^g$ to Fritz, the truth value 1 comes out for situations $g(\boldsymbol{i})$ in which Eike (= $g(\boldsymbol{x})$) is meeting Fritz; $\|(\boldsymbol{\lambda y.\mathbf{T}(i)(y)(y)})\|^g$ only yields this truth value when applied to Fritz in the (remote) situations $g(\boldsymbol{i})$, in which Fritz meets himself.

The counter-example turns on the fact that the variable $\boldsymbol{y}$ gets into the scope of a '$\boldsymbol{\lambda y}$' and thus bound when inserted for $\boldsymbol{x}$: since binding creates assignment-independence, the fact that $\boldsymbol{x}$ and $\boldsymbol{y}$ co-designate given the assignment g, becomes irrelevant; the bound $\boldsymbol{y}$ has no independent denotation. In particular, the denotational equivalence of the sub-expressions $\boldsymbol{x}$ and $\boldsymbol{y}$ cannot carry over to the whole formula. As a consequence, the principle (58) must be restricted so that formerly free variables $\boldsymbol{z}$ must not be 'accidentally' bound – i.e., get into the scope of a '$\boldsymbol{\lambda z}$' – when they replace the variable $\boldsymbol{x}$. The risk of accidental binding not only arises where the formula $\boldsymbol{\delta}$ to be substituted is itself a variable but also if $\boldsymbol{\delta}$ is a complex formula containing a free variable that would get bound after substitution; a pertinent example will be scrutinised in the exercise section. In order to formulate the principle intended in (58) in a reasonably transparent way, a special term for the fact that a variable gets accidentally bound, is advisable:

(60) *Definition*
Let $\boldsymbol{\alpha}$ and $\boldsymbol{\delta}$ be type-logical formulae and let $\boldsymbol{x}$ be a variable of the same type as $\boldsymbol{\delta}$. Then $\boldsymbol{x}$ is *free for [substitution of] $\boldsymbol{\delta}$ in $\boldsymbol{\alpha}$* if there is no $\boldsymbol{z} \in Fr(\boldsymbol{\delta})$ such that $\boldsymbol{\alpha}$ contains a sub-formula of the form '$(\boldsymbol{\lambda z.\beta})$' such that $\boldsymbol{x} \in Fr(\boldsymbol{\lambda z.\beta})$.

The above definition becomes clearer if one considers the circumstances in which a variable $\boldsymbol{x}$ is *not free* for a formula $\boldsymbol{\delta}$ within a formula $\boldsymbol{\alpha}$. According to (60), this is precisely the case if $\boldsymbol{\delta}$ contains a free variable $\boldsymbol{z}$ and there is a part '$(\boldsymbol{\lambda z.\ \beta})$' in $\boldsymbol{\alpha}$ in which $\boldsymbol{x}$ is free. If one then replaces (the free occurrences of) $\boldsymbol{x}$ in $\boldsymbol{\alpha}$ by $\boldsymbol{\delta}$s, one would, in particular, have to replace $\boldsymbol{x}$ in '$(\boldsymbol{\lambda z.\ \beta})$'. But then the newly inserted $\boldsymbol{\delta}$ would end up in the scope of '$\boldsymbol{\lambda z}$' – and with it the free occurrences of $\boldsymbol{z}$ in $\boldsymbol{\delta}$ – which would thus be accidentally bound. This is precisely what happens if $\boldsymbol{x}$ is *not* free for $\boldsymbol{\delta}$ in $\boldsymbol{\alpha}$; and this is

precisely what needs to be excluded. We thus obtain the following restricted, but correct version of (58):

(61) *Substitution Lemma*
Let $\boldsymbol{\alpha}$ and $\boldsymbol{\delta}$ be type-logical formulae and let $\boldsymbol{x}$ be a variable (of the same type as $\boldsymbol{\delta}$) which is free for $\boldsymbol{\delta}$ in $\boldsymbol{\alpha}$. Then for any assignment g the following holds:
If $\|\boldsymbol{x}\|^g = \|\boldsymbol{\delta}\|^g$, then $\|\boldsymbol{\alpha}\|^g = \|\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{\delta}]\|^g$.

We skip the proof, which can again be given by proceeding inductively from simple to ever more complex formulae. Before moving on, a possible source of confusion regarding the range of the Substitution Lemma ought to be eliminated. The Substitution Lemma concerns *sameness of denotation*, not *logical equivalence*. As pointed out at the end of Section 5.3, the latter behaves fully compositionally: logically equivalent formulae may be substituted for one another *wherever they occur* – and, in particular, no matter whether they contain free variables that may get bound in the process. Moreover, the result of such a substitution will be logically equivalent to the original formula and *a fortiori* share its denotation at *any* given assignment. The side condition blocking accidental binding only concerns substitution of *denotationally equivalent* formulae, i.e., formulae that happen to have the same denotation at *some* given assignment. Here is a case in point. The formulae (62a) and (62b) are logically equivalent, given the above law (51) of the commutativity of conjunction. As a consequence, (62a) can be replaced by (62b) in (63a), with a logically equivalent result (63b), even though the free $\boldsymbol{x}$ in (62b) gets bound during this substitution:

(62) a. $[\boldsymbol{P}^{\boldsymbol{et}}(\boldsymbol{x}^{\boldsymbol{e}}) \wedge \boldsymbol{Q}^{\boldsymbol{et}}(\boldsymbol{x})]$
b. $[\boldsymbol{Q}^{\boldsymbol{et}}(\boldsymbol{x}^{\boldsymbol{e}}) \wedge \boldsymbol{P}^{\boldsymbol{et}}(\boldsymbol{x})]$

(63) a. $(\boldsymbol{\lambda x^e.}\ [\boldsymbol{P}^{\boldsymbol{et}}(\boldsymbol{x}) \wedge \boldsymbol{Q}^{\boldsymbol{et}}(\boldsymbol{x})])$
b. $(\boldsymbol{\lambda x^e.}\ [\boldsymbol{Q}^{\boldsymbol{et}}(\boldsymbol{x}) \wedge \boldsymbol{P}^{\boldsymbol{et}}(\boldsymbol{x})])$

The fact that logically equivalent formulae can be freely substituted in any environment will be of great help in the reduction of type-logical translations of German expressions to be explored from Section 5.5 onward.

Let us now take a look at a consequence jointly implied by the Substitution Lemma and the Coincidence Lemma, viz. the arbitrariness of variable names, already mentioned above. Basically, this arbitrariness amounts to the possibility of rewriting any binding of the form $(\boldsymbol{\lambda x.}\ \ldots)$ as $(\boldsymbol{\lambda y.}\ \ldots)$, provided that all lambda-bound $\boldsymbol{x}$ in '…' are replaced by $\boldsymbol{y}$. However, this $\boldsymbol{y}$ must not already be free in '…'; otherwise the $(\boldsymbol{\lambda y.}\ \ldots)$ would bind these free occurrences. (The formula $(\boldsymbol{\lambda x.}\ \boldsymbol{x = y})$ may serve as an illustration – but only in an exercise.) Moreover, accidental binding by substitutions must again be prevented; $\boldsymbol{x}$ thus needs to be free for $\boldsymbol{y}$.

(64) *Principle of bound renaming*
Let $\boldsymbol{\alpha}$ be a type-logical formula and let $\boldsymbol{x}$ be a variable. Then if $\boldsymbol{y}$ is a variable (of the same type as $\boldsymbol{x}$) that is not free in $\boldsymbol{\alpha}$ and for which $\boldsymbol{x}$ is free in $\boldsymbol{\alpha}$, the following holds:

$$(\boldsymbol{\lambda x.\ \alpha}) \equiv (\boldsymbol{\lambda y.\ \alpha}[\boldsymbol{x}/\boldsymbol{y}]).$$

To see that (64) is correct on the basis of the interpretation of lambda-abstraction given in Section 5.3, one may consider any assignment and show that the denotations of the two formulae in (64) have the same denotation (in the circumstances defined there). Since lambda-abstraction always results in formulae of types ab, these denotations are always functions from objects of the type a of the variables $\boldsymbol{x}$ and $\boldsymbol{y}$ to objects of the type b of $\boldsymbol{\alpha}$. And such functions are identical if they yield the same value whenever applied to any object u of type a. So it must be showen that $\|(\boldsymbol{\lambda x.\ \alpha})\|^g(u) = \|(\boldsymbol{\lambda y.\ \alpha}[\boldsymbol{x}/\boldsymbol{y}])\|^g(u)$. And this is indeed the case:

(65)	$\|(\boldsymbol{\lambda x.\alpha})\|^g(u)$	by the interpretation (47) of abstraction
=	$\|\boldsymbol{\alpha}\|^{g[\boldsymbol{x}/u]}$	by the Coincidence Lemma: $\boldsymbol{y} \notin Fr(\boldsymbol{\alpha})$
=	$\|\boldsymbol{\alpha}\|^{g[\boldsymbol{x}/u][\boldsymbol{y}/u]}$	by the Substitution Lemma: $\boldsymbol{x}$ is free for $\boldsymbol{y}$ in $\boldsymbol{\alpha}$, and $\|\boldsymbol{x}\|^{g[\boldsymbol{x}/u][\boldsymbol{y}/u]} = u = \|\boldsymbol{y}\|^{g[\boldsymbol{x}/u][\boldsymbol{y}/u]}$
=	$\|\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{y}]\|^{g[\boldsymbol{x}/u][\boldsymbol{y}/u]}$	by the Coincidence Lemma: $\boldsymbol{x} \notin Fr(\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{y}])$
=	$\|\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{y}]\|^{g[\boldsymbol{y}/u]}$	by the interpretation (47) of abstraction
=	$\|(\boldsymbol{\lambda y.\alpha}[\boldsymbol{x}/\boldsymbol{y}])\|^g(u)$	

The logical equivalence given in (64) is a transformation that will be frequently applied in indirect interpretation. It will turn out to be crucial that there are always enough variables $\boldsymbol{y}$ that meet the conditions imposed in (64) – $\boldsymbol{x}$ is free for them and they do not occur in the scope of '$\boldsymbol{\lambda x}$'; indeed, there are infinitely many variables of any type.

Clearly the most important logical transformation is lambda-conversion, which we have already encountered in the meta-language. As a type-logical equivalence it can be given a much more precise formulation though:

(66) *Principle of $\boldsymbol{\lambda}$-conversion*
Let $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ be type-logical formulae and let $\boldsymbol{x}$ be a variable (of the same type as $\boldsymbol{\beta}$) that is free for $\boldsymbol{\beta}$ in $\boldsymbol{\alpha}$. Then the following holds:

$$(\boldsymbol{\lambda x.\alpha})(\boldsymbol{\beta}) \equiv \boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{\beta}].$$

Of course, $\boldsymbol{\lambda}$-conversion needs to shun accidental binding too. As a case in point, $(\boldsymbol{\lambda x.}(\exists \boldsymbol{y})\mathbf{T}(\boldsymbol{i})(\boldsymbol{x})(\boldsymbol{y}))(\boldsymbol{y})$ is not equivalent to $(\exists \boldsymbol{y})\mathbf{T}(\boldsymbol{i})(\boldsymbol{y})(\boldsymbol{y})$, as one can easily verify (again in an exercise). The side condition on the variable excludes such cases.

(66) is basically a consequence of the Coincidence and Substitution Lemmas;

but the Principe (64) of bound renaming also plays a part. Let us take a closer look at the proof of the equivalence claimed in (66) (under the circumstances mentioned there)! First of all, in view of the interpretation of (35) of application, the following holds for all assignments g:

$$(67) \qquad \|(\boldsymbol{\lambda x.\alpha})(\boldsymbol{\beta})\|^g = \|(\boldsymbol{\lambda x.\alpha})\|^g(\|\boldsymbol{\beta}\|^g)$$

In order to determine the value to the right of the equality sign, a bound renaming is helpful. We thus pick a variable $\boldsymbol{z}$ of the type common to $\boldsymbol{x}$ and $\boldsymbol{\beta}$, that does not occur in either $\boldsymbol{\alpha}$ or $\boldsymbol{\beta}$ – neither freely nor as a bound variable nor in a lambda-prefix. There must be such $\boldsymbol{z}$; after all, there are infinitely many variables of any type. Since $\boldsymbol{z}$ does not occur in $\boldsymbol{\alpha}$, it cannot be accidentally bound when it is substituted for $\boldsymbol{x}$, which means that $\boldsymbol{x}$ is free for $\boldsymbol{z}$ in $\boldsymbol{\alpha}$; moreover, since $\boldsymbol{z}$ does not occur in $\boldsymbol{\alpha}$, the conditions for a bound renaming in (64) obtain, and so the equation (67) can be continued as follows:

$$\begin{aligned} (68) \qquad & \|(\boldsymbol{\lambda x.\alpha})(\boldsymbol{\beta})\|^g = \|(\boldsymbol{\lambda x.\alpha})\|^g(\|\boldsymbol{\beta}\|^g) = \|(\boldsymbol{\lambda z.\alpha}[\boldsymbol{x}/\boldsymbol{z}])\|^g(\|\boldsymbol{\beta}\|^g) \\ = \qquad & \|\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{z}]\|^{g[\boldsymbol{z}/\|\boldsymbol{\beta}\|^g]} \end{aligned}$$

The last transition is a simple application of the interpretation (47) of abstraction. In order to apply the Substitution Lemma now, we need to make sure that (i) $\boldsymbol{z}$ is free for $\boldsymbol{\beta}$ in $\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{z}]$ and (ii) $\boldsymbol{z}$ and $\boldsymbol{\beta}$ have the same denotation given the assignment $g[\boldsymbol{z}/\|\boldsymbol{\beta}\|^g]$. (i) holds because $\boldsymbol{z}$ occupies precisely those positions in $\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{z}]$ at which $\boldsymbol{x}$ stood in $\boldsymbol{\alpha}$; and no variables from $\boldsymbol{\beta}$ gets accidentally bound in these positions – for the general assumptions in (66) say that $\boldsymbol{x}$ is free für $\boldsymbol{\beta}$ in $\boldsymbol{\alpha}$. But (ii) also holds: $\|\boldsymbol{z}\|^{g[\boldsymbol{z}/\|\boldsymbol{\beta}\|^g]} = g[\boldsymbol{z}/\|\boldsymbol{\beta}\|^g](\boldsymbol{z}) = \|\boldsymbol{\beta}\|^g = \|\boldsymbol{\beta}\|^{g[\boldsymbol{z}/\|\boldsymbol{\beta}\|^g]}$ – due to the Coincidence Lemma and because $\boldsymbol{z} \notin Fr(\boldsymbol{\beta})$. We may thus apply the Substitution Lemma, turning (68) into:

$$(69) \qquad \|(\boldsymbol{\lambda x.\alpha})(\boldsymbol{\beta})\|^g = \cdots = \|\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{z}]\|^{g[\boldsymbol{z}/\|\boldsymbol{\beta}\|^g]} = \|\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{z}][\boldsymbol{z}/\boldsymbol{\beta}]\|^{g[\boldsymbol{z}/\|\boldsymbol{\beta}\|^g]}$$

But then $\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{z}][\boldsymbol{z}/\boldsymbol{\beta}] = \boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{\beta}]$: the provisionally inserted $\boldsymbol{z}$ stand where we originally had an $\boldsymbol{x}$ – and, by assumption, no other $\boldsymbol{z}$ occur in $\boldsymbol{\alpha}$.[19] (69) thus reduces to:

$$(70) \qquad \|(\boldsymbol{\lambda x.\alpha})(\boldsymbol{\beta})\|^g = \cdots = \|\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{\beta}]\|^{g[\boldsymbol{z}/\|\boldsymbol{\beta}\|^g]}$$

Given that $\boldsymbol{z}$ did not occur in $\boldsymbol{\alpha}$ or $\boldsymbol{\beta}$, it cannot occur in $\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{\beta}]$ either; and since g and $g[\boldsymbol{z}/\|\boldsymbol{\beta}\|^g]$ only differ in the value for this $\boldsymbol{z}$, $\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{\beta}]$ has the same denotation at these two assignments, due to the Coincidence Lemma.

[19] More generally, and more precisely speaking, for any formualae $\boldsymbol{\beta}$ and variables $\boldsymbol{x}$ and $\boldsymbol{z}$ of the same type the following holds: if $\boldsymbol{x}$ is free for $\boldsymbol{z}$ in any formula $\boldsymbol{\alpha}$ und $\boldsymbol{z} \notin Fr(\boldsymbol{\alpha})$, then: $\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{z}][\boldsymbol{z}/\boldsymbol{\beta}] = \boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{\beta}]$. Again a stepwise (inductive) proof can be given for this claim.

(70) thus boils down to the required equivalence from the principle (66) of $\boldsymbol{\lambda}$-conversion:

(71) $\|(\boldsymbol{\lambda x.\alpha})(\boldsymbol{\beta})\|^g = \cdots = \|\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{\beta}]\|^{g[\boldsymbol{z}/\|\boldsymbol{\beta}\|^g]} = \|\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{\beta}]\|^g$

In practice, $\boldsymbol{\lambda}$-conversion is chiefly applied going from left to right, i.e., to *reduce* a formula of the form $(\boldsymbol{\lambda x.\alpha})(\boldsymbol{\beta})$. The reduced formula is usually shorter and, more to the point, easier to read. Since the relevant constellation frequently arises in indirect interpretation when the side condition does not obtain, it is important to realise that the latter can always be circumvented by applications of the principle (64) of bound renaming. For should the argument $\boldsymbol{\beta}$ contain a variable $\boldsymbol{y}$ that would be accidentally bound when replacing $\boldsymbol{x}$ in $\boldsymbol{\alpha}$, *bound* $\boldsymbol{y}$ *in* $\boldsymbol{\alpha}$ may just be renamed – to wit, by a variable that does not occur in $\boldsymbol{\alpha}$ or $\boldsymbol{\beta}$ and thus bears no risk. Since there are infinitely many such variables, such renamings are always possible – if need be more than once. The next section will already offer several examples to study the alternation of bound renaming and $\boldsymbol{\lambda}$-conversion.

The proof of the law of $\boldsymbol{\lambda}$-conversion (66) might appear rather complicated; however, its purpose is to make sure that this equivalence is not an arbitrary formal stipulation but a transformation that is fully justified by the interpretation of type-logical formulae given in the previous section. Despite appearances to the contrary, the formulae do not owe their meanings to the reduction process, but to their compositional interpretation. Reductions preserve the meanings of formulae; however, they usually make them easier to grasp.

One special case of $\boldsymbol{\lambda}$-conversion is particularly frequent and important, viz. when the argument $\boldsymbol{\beta}$ in the constellation $(\boldsymbol{\lambda x.\alpha})(\boldsymbol{\beta})$ happens to be the variable bound in the prefix: $(\boldsymbol{\lambda x.\alpha})(\boldsymbol{x})$. It may not be entirely obvious that the side condition on the variable is satisfied in this case; for the argument $\boldsymbol{\beta}$ is the variable $\boldsymbol{x}$, which means that $Fr(\boldsymbol{\beta}) \neq \emptyset$. In order to apply $\boldsymbol{\lambda}$-conversion one thus first needs to check whether a variable free in $\boldsymbol{\beta}$ gets accidentally bound when inserted into $\boldsymbol{\alpha}$. Since $Fr(\boldsymbol{\beta}) = \{\boldsymbol{x}\}$, it suffices to check this for $\boldsymbol{x}$ – hence whether the variable $\boldsymbol{x}$ bound by the $\boldsymbol{\lambda}$ is free for $\boldsymbol{x}$ itself in $\boldsymbol{\alpha}$. But then the substitution process captures only the free occurrences of $\boldsymbol{x}$ in $\boldsymbol{\alpha}$ and if these free $\boldsymbol{x}$ are again replaced by $\boldsymbol{x}$, obviously nothing happens – $\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{x}] = \boldsymbol{\alpha}$; in particular, the $\boldsymbol{x}$ remain free, which excludes any accidental binding. So the conversion may be carried out, and since $\boldsymbol{\alpha}[\boldsymbol{x}/\boldsymbol{x}] = \boldsymbol{\alpha}$ it boils down to dropping both the prefix '$\boldsymbol{\lambda x.}$' and the argument '$(\boldsymbol{x})$'. We reserve a special principle for this particular case of $\boldsymbol{\lambda}$-conversion:

(72) *Principle of* eigen-*conversion*
Let $\boldsymbol{\alpha}$ be a type-logical formula and let $\boldsymbol{x}$ be a variable. Then the following holds:

$$(\boldsymbol{\lambda x.\alpha})(\boldsymbol{x}) \equiv \boldsymbol{\alpha}.$$

Apart from the fundamental equivalences introduced so far, there are a few further logical transformations, which – though not as frequently called upon – may occasionally be of service and are certainly essential for a full understanding of the formalism. Some of them are listed below; further ones will be mentioned when we make use of them. They are all quite easy to establish:

(73) For all variables $\boldsymbol{x}$ and $\boldsymbol{y}$ of the same type, all formulae $\boldsymbol{\varphi}$, $\boldsymbol{\psi}$ and $\boldsymbol{\chi}$ of type $\boldsymbol{t}$, and all formulae $\boldsymbol{\alpha}$ of any type, the following holds:

a. $\neg\neg\boldsymbol{\varphi} \equiv \boldsymbol{\varphi}$ *Law of Double Negation*

b. $[\boldsymbol{\varphi} \wedge [\boldsymbol{\psi} \wedge \boldsymbol{\chi}]] \equiv [[\boldsymbol{\varphi} \wedge \boldsymbol{\psi}] \wedge \boldsymbol{\chi}]$ *associativity of conjunction*

c. $(\exists \boldsymbol{x})(\exists \boldsymbol{y})\boldsymbol{\varphi} \equiv (\exists \boldsymbol{y})(\exists \boldsymbol{x})\boldsymbol{\varphi}$

d. $(\exists \boldsymbol{x})[\boldsymbol{\varphi} \wedge (\exists \boldsymbol{y})\boldsymbol{\psi}] \equiv (\exists \boldsymbol{x})(\exists \boldsymbol{y})[\boldsymbol{\varphi} \wedge \boldsymbol{\psi}]$ – *if* $\boldsymbol{y} \notin Fr(\boldsymbol{\varphi})$

e. $(\boldsymbol{\lambda x.\alpha(x)}) \equiv \boldsymbol{\alpha}$ – *if* $\boldsymbol{x} \notin Fr(\boldsymbol{\alpha})$ *[η-conversion]*

5.5 Indirect interpretation

We are now in a position to reconstruct the interpretation of natural language constructions developed in the previous chapters within the framework of indirect interpretation. We will proceed by defining, for each natural language expression A, a type-logical translation $|A|$ whose denotation is precisely the extension of A according to the previously given direct interpretation. Given our preparatory remarks, this is not particularly difficult. We proceed compositionally and thus need to first give type-logical counterparts for the lexical expressions. In most cases these will take the form $\mathbf{c(\boldsymbol{i})}$, which we will, from now on, abbreviate as $\mathbf{c}_{\boldsymbol{i}}$:

(74) $|\mathbf{Eike}| = \mathbf{e}$; $|\mathbf{Fritz}| = \mathbf{f}$; $|\mathbf{Hans}| = \mathbf{h}$; ... [all of them constants of type $\boldsymbol{e}$]

$|\mathbf{arbeitet}| = \mathbf{A}_{\boldsymbol{i}}$; $|\mathbf{Mann}| = \mathbf{M}_{\boldsymbol{i}}$; $|\mathbf{Frau}| = \mathbf{F}_{\boldsymbol{i}}$; $|\mathbf{Porsche}| = \mathbf{P}_{\boldsymbol{i}}$; ...

[where $\mathbf{A}$, $\mathbf{M}$, $\mathbf{F}$, $\mathbf{P}$, ... are constants of type $\boldsymbol{s(et)}$]

$|\mathbf{trifft}| = \mathbf{T}_{\boldsymbol{i}}$; $|\mathbf{besitzt}| = \mathbf{B}_{\boldsymbol{i}}$; $|\mathbf{sieht}| = \mathbf{S}_{\boldsymbol{i}}$; $|\mathbf{knutscht}| = \mathbf{K}_{\boldsymbol{i}}$; ...

[where $\mathbf{T}$, $\mathbf{B}$, $\mathbf{S}$, $\mathbf{K}$, ... are constants of type $\boldsymbol{s(e(et))}$]

Further lexical translations will be introduced when need be; we will then take it for granted that the constants are interpreted like the corresponding lexical expressions in direct interpretation: the (assignment-independent) denotation of $\mathbf{f}$ is the individual Fritz; the denotation of $\mathbf{A}$ assigns to any situation $s \in LS$ the characteristic function of individuals that work in s; etc.

The translations of the (clausal) coordinating conjunctions **und** and **oder** draw on the above logical constant $\wedge$, which translates the former and is part of the more complex translation of the latter:

(75) a. $|\mathbf{und}| = \wedge$
b. $|\mathbf{oder}| = (\boldsymbol{\lambda q^t.(\lambda p^t.\neg[\neg p \wedge \neg q])})$

Successive application the denotation of (75b) to two truth values u and v yields the value 1 if it is not the case that $u = v = 0$ – i.e. just in case at least one of the two values is 1. We could, of course, also instead have introduced a logical constant $\vee$ translating **oder** and interpret it by the truth table in (8) in Section 5.1. Yet on the one hand, the fact that disjunction can be reduced to conjunction (and negation) is of intrinsic interest.[20] On the other hand, we may as well read the notation $[\varphi \vee \psi]$ as short for the combination $\neg[\neg\varphi \wedge \neg\psi]$ of formulae used in (75b) – which is what we will do from now on.

Most determiners will be translated by complex formulae as well. We have already come across two examples, viz. the translations of **ein-** and **kein-** in (25) und (26). The other ones are rather straightforward, given the direct interpretation in Section 3.2:

(76) a. $|\mathbf{kein\text{-}}| = (\boldsymbol{\lambda Q^{et}.(\lambda P^{et}.\neg(\exists x^e)[Q(x) \wedge P(x)])})$
b. $|\mathbf{ein\text{-}}_{indef}| = (\boldsymbol{\lambda Q^{et}.(\lambda P^{et}.(\exists x^e)[Q(x) \wedge P(x)])})$
c. $|\mathbf{ein\text{-}}_{Num}| = (\boldsymbol{\lambda Q^{et}.(\lambda P^{et}.(\exists x^e)[Q(x) \wedge P(x) \wedge \neg(\exists y^e)[\neg(x = y) \wedge Q(y) \wedge P(y)]])})$
d. $|\mathbf{jed\text{-}}| = (\boldsymbol{\lambda Q^{et}.(\lambda P^{et}.\neg(\exists x^e)[Q(x) \wedge \neg P(x)])})$
e. $|\mathbf{die\ meisten}| = \mathbf{MOST}$ [constant of type *(et)((et)t)*]
f. $|\mathbf{d\text{-}}_{Russell}| = (\boldsymbol{\lambda Q^{et}.(\lambda P^{et}.(\exists x^e)[Q(x) \wedge \neg(\exists y^e)[\neg(x = y) \wedge Q(y)] \wedge P(x)])})$

The translation (76c) of **ein-** as a numeral looks rather complicated but only has the effect that after combining it with noun and predicate, the truth value 1 ensues if the intersection of their extensions (taken as sets) contain precisely one individual x – so that there is precisely one x that is both in the noun extension and in the predicate extension without there being a further individual y – *distinct from* x, that is – in this intersection. Moreover, it is not hard to see that the denotations of the translations (76d) and (76f) also coincide with the corresponding interpretations given in Chapter 3. Finally, as to (76e), we merely translated the determinator **die meisten** by a non-logical constant which we take to be interpreted accordingly.

Let us move on to complex expressions. Following the Principle of Compositionality, we will, for each grammatical construction discussed in the preceding chapters, specify how the type-logical translation of the constructed expression is obtained from the translations of its parts. We start with the

[20] The reduction, which also works in the opposite direction, is called *de Morgan's Law*, in honour of the English logician Augustus de Morgan (1806–1871), although it had already been known in classical antiquity.

coordination of clauses analysed in Chapter 1; in indirect interpretation it comes out as follows:

(77) *Indirect interpretation of clausal coordination*
If S and S' are (declarative) sentences and K is a coordinating conjunction, the following holds:
$$|S\ K\ S'| = |K|(|S'|)(|S|).$$

It is worth stopping to note how the equation in (77) is to be understood. According to it, the translation of two sentences S und S' coordinated by **und** or **oder** (and more generally, by a conjunction K) – i.e. the translation $|S\ K\ S'|$ of '$S\ K\ S'$' – arises from the type-logical translation of the conjunction itself (i.e., some formula $|K|$), followed by the translations of the coordinated clauses (i.e., certain formulae $|S'|$ and $|S|$, each between (boldface) parentheses. Since $|S|$ and $|S'|$ are themselves formulae of type $\boldsymbol{t}$ and conjunctions K translate as formulae of type $\boldsymbol{t(tt)}$, the end result of (77) is a formula of type $\boldsymbol{t}$. Before applying the rule to a specific example, we need to translate the constructions from Chapter 2:

(78) *Indirect interpretation of predication*
If S is a sentence whose subject is a proper name N and whose predicate is P, the following holds:
$$|S| = |P|(|N|).$$

(79) *Indirect interpretation of names as direct objects*
If P is a predicate consisting of a verb V and a proper name N as its object, the following holds:
$$|P| = |V|(|N|).$$

Let us test these rules on an example:

(80) **Eike arbeitet und Fritz trifft Hans.**
[≈ Eike is working and Fritz is meeting Hans.]

The translation follows the obvious constituent structure of the sentence:

(81)
$$\begin{aligned}
&|\mathbf{Eike\ arbeitet\ und\ Fritz\ trifft\ Hans}| && \text{by (77)}\\
=\ &|\mathbf{und}|(|\mathbf{Fritz\ trifft\ Hans}|)(|\mathbf{Eike\ arbeitet}|) && \text{by (78) [2×]}\\
=\ &|\mathbf{und}|(|\mathbf{trifft\ Hans}|(|\mathbf{Fritz}|))(|\mathbf{arbeitet}|(|\mathbf{Eike}|)) && \text{by (79)}\\
=\ &|\mathbf{und}|(|\mathbf{trifft}|(|\mathbf{Hans}|)(|\mathbf{Fritz}|))(|\mathbf{arbeitet}|(|\mathbf{Eike}|)) && \text{by (74) and (75a)}\\
=\ &\wedge(\mathbf{T}_i(\mathbf{h})(\mathbf{f}))(\mathbf{A}_i(\mathbf{e}))
\end{aligned}$$

The resulting formula can be rewritten using the notation used for conjunction as $[\mathbf{A}_i(\mathbf{e}) \wedge \mathbf{T}_i(\mathbf{h})(\mathbf{f})]$; using a similar convention, the right conjunct can be brought into (German and English) surface order, thus obtaining:

(82) $\mathbf{[A_{\mathit{i}}(e) \wedge T_{\mathit{i}}(f,h)]}$

It should be noted that the reformulation (82) of the translation (81) determined in (80) is merely a notational variant. The type-logical formula is exactly the same; it is merely represented differently. Thus the transition is not a logical reduction based on any transformation laws. Indeed, logical transformations can only make this formula less readable. The case of the following variation is different:

(83) **Eike arbeitet oder Fritz trifft Hans.**

The translation obviously proceeds as in (81) – with the exception of the final line, where the complex translation (75b) of **oder** takes the place of the logical constant $\wedge$:

(84)	$\mathbf{\lvert Eike\ arbeitet\ oder\ Fritz\ trifft\ Hans\rvert}$	by (77)
=	$\mathbf{\lvert oder\rvert(\lvert Fritz\ trifft\ Hans\rvert)(\lvert Eike\ arbeitet\rvert)}$	by (78) [2×]
=	$\mathbf{\lvert oder\rvert(\lvert trifft\ Hans\rvert(\lvert Fritz\rvert))(\lvert arbeitet\rvert(\lvert Eike\rvert))}$	by (79)
=	$\mathbf{\lvert oder\rvert(\lvert trifft\rvert(\lvert Hans\rvert)(\lvert Fritz\rvert))(\lvert arbeitet\rvert(\lvert Eike\rvert))}$	by (74) and (75b)
=	$\mathbf{(\lambda q.(\lambda p.[p \vee q]))(T_{\mathit{i}}(h)(f))(A_{\mathit{i}}(e))}$	

Again we can apply the convention just introduced to repackage the translation of the second clause – but to no avail:

(85) $\mathbf{(\lambda q.(\lambda p.[p \vee q]))(T_{\mathit{i}}(f,h))(A_{\mathit{i}}(e))}$

But of course, this formula can be further reduced, even though the formula as a whole does not have the form required by (66): it is an application $\boldsymbol{\gamma(\delta)}$, where $\boldsymbol{\gamma}$ is not a $\boldsymbol{\lambda}$-term, but again an application. However this part $\boldsymbol{\gamma}$ constitutes a constellation relevant for $\boldsymbol{\lambda}$-conversion: it is of the form '$\mathbf{(\lambda x.\alpha)(\beta)}$' – where $\boldsymbol{\alpha}$ is the part starting with '$\mathbf{(\lambda q.}\ldots$' and ending before '$\mathbf{(T_{\mathit{i}}(f,h))}$', $\boldsymbol{x}$ is $\boldsymbol{q}$, and $\boldsymbol{\beta}$ is the argument $\mathbf{T_{\mathit{i}}(f,h)}$. Since this $\boldsymbol{\beta}$ contains no free variables (apart from $\boldsymbol{i}$), $\boldsymbol{q}$ is free for $\boldsymbol{\beta}$ in $\boldsymbol{\alpha}$, and the conversion can proceed:

(86)	$\mathbf{\lvert Eike\ arbeitet\ oder\ Fritz\ trifft\ Hans\rvert}$	
=	...	using (84)
=	$\mathbf{(\lambda q.(\lambda p.[p \vee q]))(T_{\mathit{i}}(h)(f))(A_{\mathit{i}}(e))}$	
=	$\mathbf{(\lambda q.(\lambda p.[p \vee q]))(T_{\mathit{i}}(f,h))(A_{\mathit{i}}(e))}$	= (85)
≡	$\mathbf{(\lambda p.[p \vee T_{\mathit{i}}(f,h)])(A_{\mathit{i}}(e))}$	

– which is itself again a relevant constellation for a further conversion:

(87)	$\mathbf{\lvert Eike\ arbeitet\ oder\ Fritz\ trifft\ Hans\rvert}$	
	...	
≡	$\mathbf{[A_{\mathit{i}}(e) \vee T_{\mathit{i}}(f,h)]}$	

It should be noted that the last two transitions in (86) and (87) are no equations (=); the application of $\boldsymbol{\lambda}$-conversion (and logical transformations in general) leads to new, equivalent formulae ($\equiv$).

(87) is a (logically) reduced translation of (83), i.e. a formula that is logically equivalent to the real translation in (84). For two reasons, the reduction of (85) to (87) is relatively simple. For one thing, (a) at each step there was only one possibility of applying the law of $\boldsymbol{\lambda}$-conversion; both the translation (85) itself and the first reduction (86) only contain one relevant constellation. For another thing, (b) the side condition on variables obtained in both cases, since the argument to be inserted contained no free variables, apart from $\boldsymbol{i}$, which never gets bound anyway (so far). However, (a) and (b) are rather the exception than the rule, as we will see once the remaining constructions have been translated and made to interact. These constructions introduce quantifying nominals in subject and object position. In the case of subject quantification, functor and argument merely get reversed, vis-à-vis the translation (78) of predication:

(88) *Indirect interpretation of quantification*
If S is a sentence whose subject position is occupied by a quantificational noun phrase Q and whose predicate is P, the following holds:

$$|S| = |Q|(|P|).$$

In the case of quantifying objects, too, the interpretation given in Section 3.4 immediately carries over to type-logical formulae:

(89) *Indirect interpretation of quantificational noun phrases in object position*
If P is a predicate consisting of a verb V and a quantificational noun phrase Q as its object, the following holds:[21]

$$|P| = \boldsymbol{\lambda x^e.}|Q|((\boldsymbol{\lambda y^e.}|V|(\boldsymbol{y})(\boldsymbol{x}))).$$

We will apply (88) and (89) to an example in due course, thus exploring the interaction of bound renaming and $\boldsymbol{\lambda}$-conversion. However, before that, we need to specify the translation of non-lexical noun phrases:

(90) *Indirect interpretation of quantificational noun phrases*
If QN is a quantificational noun phrase consisting of a determiner D and a noun N, the following holds:

$$|QN| = |D|(|N|).$$

[21] The double parentheses around the part starting with '$\boldsymbol{\lambda x.}$' are due to the syntax of type logic: according to *(Abs)*, the lambda term starts and ends with a bracket; and, being an argument of 'Q', it has to be surrounded by brackets again – by *(App)*. We will, however, usually omit these double brackets in the future.

With (90), we have now covered the interpretation given in Chapters 1–3 within the methodology of indirect interpretation. Let us now look at a somewhat more complex example:

(91) **Jeder Mann trifft eine Frau.**

This is what the relevant translation rules produce:

(92)
$|\textbf{Jeder Mann trifft eine Frau}|$ by (88)
$= |\textbf{Jeder Mann}|(|\textbf{trifft eine Frau}|)$ by (90)
$= |\textbf{Jeder}|(|\textbf{Mann}|)(|\textbf{trifft eine Frau}|)$ by (89)
$= |\textbf{Jeder}|(|\textbf{Mann}|)(\boldsymbol{\lambda x^e}.|\textbf{eine Frau}|(\boldsymbol{\lambda y^e}.|\textbf{trifft}|(\boldsymbol{x},\boldsymbol{y})))$ by (90)
$= |\textbf{Jeder}|(|\textbf{Mann}|)(\boldsymbol{\lambda x}.|\textbf{eine}|(|\textbf{Frau}|)(\boldsymbol{\lambda y}.|\textbf{trifft}|(\boldsymbol{x},\boldsymbol{y})))$ by (74) [3×]
$= |\textbf{Jeder}|(\mathbf{M}_i)(\boldsymbol{\lambda x}.|\textbf{eine}|(\mathbf{F}_i)(\boldsymbol{\lambda y}.\mathbf{T}_i(\boldsymbol{x},\boldsymbol{y})))$ by (76d)
$= (\boldsymbol{\lambda Q^{et}}.(\boldsymbol{\lambda P^{et}}.\neg(\exists \boldsymbol{x})[\boldsymbol{Q}(\boldsymbol{x}) \wedge \neg \boldsymbol{P}(\boldsymbol{x})]))(\mathbf{M}_i)$
$(\boldsymbol{\lambda x}.|\textbf{eine}|(\mathbf{F}_i)(\boldsymbol{\lambda y}.\mathbf{T}_i(\boldsymbol{x},\boldsymbol{y})))$ by (76b)
$= (\boldsymbol{\lambda Q}.(\boldsymbol{\lambda P}.\neg(\exists \boldsymbol{x})[\boldsymbol{Q}(\boldsymbol{x}) \wedge \neg \boldsymbol{P}(\boldsymbol{x})]))(\mathbf{M}_i)$
$(\boldsymbol{\lambda x}.(\boldsymbol{\lambda Q}.(\boldsymbol{\lambda P}.(\exists \boldsymbol{x})[\boldsymbol{Q}(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})]))(\mathbf{F}_i)(\boldsymbol{\lambda y}.\mathbf{T}_i(\boldsymbol{x},\boldsymbol{y})))$

The type-logical translation (91) determined in (92) offers a first chance to profit from the compositionality of logical equivalence: though it is not itself a pertinent constellation of the form '$(\boldsymbol{\lambda x}.\boldsymbol{\alpha})(\boldsymbol{\beta})$', it does contain two such sub-formulae to which $\boldsymbol{\lambda}$-conversion may be applied (subject to the variable condition). For one thing, the whole formula has the form '$\underline{(\boldsymbol{\lambda Q}.\boldsymbol{\alpha})(\boldsymbol{\beta})}(\boldsymbol{\gamma})$' – where conversion can be applied to the underlined part, since $\boldsymbol{\beta}$ is the formula $\mathbf{M}(\boldsymbol{i})$, in which $\boldsymbol{i}$ is the only free variable, but does not get bound anywhere in the $\boldsymbol{\alpha}$-part. For another thing, the argument $\boldsymbol{\gamma}$ is itself of the form '$(\boldsymbol{\lambda x}.\underline{(\boldsymbol{\lambda Q}.\boldsymbol{\alpha'})(\boldsymbol{\beta'})}(\boldsymbol{\gamma'}))$', where the conversion rule may target the doubly underlined part – and is again applicable, given that $\boldsymbol{\beta'}$ is the formula $\mathbf{F}_{\boldsymbol{i}}$ and $\boldsymbol{i}$ is not bound in $\boldsymbol{\alpha'}$. Reducing the singly underlined formula leads to (93a); if, on the other hand, conversion is performed on the doubly underlined part, (93b) results:

(93) a. $(\boldsymbol{\lambda P}.\neg(\exists \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x}) \wedge \neg \boldsymbol{P}(\boldsymbol{x})])$
$(\boldsymbol{\lambda x}.(\boldsymbol{\lambda Q}.(\boldsymbol{\lambda P}.(\exists \boldsymbol{x})[\boldsymbol{Q}(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})]))(\mathbf{F}_i)(\boldsymbol{\lambda y}.\mathbf{T}_i(\boldsymbol{x},\boldsymbol{y})))$

b. $(\boldsymbol{\lambda Q}.(\boldsymbol{\lambda P}.\neg(\exists \boldsymbol{x})[\boldsymbol{Q}(\boldsymbol{x}) \wedge \neg \boldsymbol{P}(\boldsymbol{x})]))(\mathbf{M}_i)$
$(\boldsymbol{\lambda x}.(\boldsymbol{\lambda P}.(\exists \boldsymbol{x})[\mathbf{F}_i(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})])(\boldsymbol{\lambda y}.\mathbf{T}_i(\boldsymbol{x},\boldsymbol{y})))$

(93a) has the form '$(\boldsymbol{\lambda P}.\boldsymbol{\alpha})(\boldsymbol{\beta})$', where the argument (as is readily seen) contains no free variable other than $\boldsymbol{i}$, which is not bound anywhere (as always). So (93a) λ-reduces to:

(94) $\neg(\exists \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x}) \wedge$
$\neg(\boldsymbol{\lambda x}.(\boldsymbol{\lambda Q}.(\boldsymbol{\lambda P}.(\exists \boldsymbol{x})[\boldsymbol{Q}(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})]))(\mathbf{F}_i)(\boldsymbol{\lambda y}.\mathbf{T}_i(\boldsymbol{x},\boldsymbol{y})))(\boldsymbol{x})]$

The argument of the second (innermost) negation in (94) can be *eigen*-reduced to obtain (95a); in a second step one may get rid of the '$\boldsymbol{\lambda Q}$' and insert the argument '$\mathbf{F}_{\boldsymbol{i}}$', thus obtaining (96). Alternatively, starting from (94), one could have gotten rid of the '$\boldsymbol{\lambda Q}$' first and then performed an *eigen*-conversion on the resulting (95b) – with the very same result, viz. (96).

(95) a. $\neg(\exists \boldsymbol{x})[\mathbf{M}_{\boldsymbol{i}}(\boldsymbol{x}) \wedge \neg(\boldsymbol{\lambda Q.}(\boldsymbol{\lambda P.}(\exists \boldsymbol{x})[\boldsymbol{Q}(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})]))(\mathbf{F}_{\boldsymbol{i}})(\boldsymbol{\lambda y.}\mathbf{T}_{\boldsymbol{i}}(\boldsymbol{x},\boldsymbol{y}))]$

b. $\neg(\exists \boldsymbol{x})[\mathbf{M}_{\boldsymbol{i}}(\boldsymbol{x}) \wedge \neg(\boldsymbol{\lambda x.}(\boldsymbol{\lambda P.}(\exists \boldsymbol{x})[\mathbf{F}_{\boldsymbol{i}}(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})])(\boldsymbol{\lambda y.}\mathbf{T}_{\boldsymbol{i}}(\boldsymbol{x},\boldsymbol{y})))(\boldsymbol{x})]$

(96) $\neg(\exists \boldsymbol{x})[\mathbf{M}_{\boldsymbol{i}}(\boldsymbol{x}) \wedge \neg(\boldsymbol{\lambda P.}(\exists \boldsymbol{x})[\mathbf{F}_{\boldsymbol{i}}(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})])(\boldsymbol{\lambda y.}\mathbf{T}_{\boldsymbol{i}}(\boldsymbol{x},\boldsymbol{y}))]$

Trying to further reduce (96) by trading the $\boldsymbol{\lambda P}$-prefix for the argument '$\boldsymbol{\lambda y.}\mathbf{T}_{\boldsymbol{i}}(\boldsymbol{x},\boldsymbol{y})$', leads to a conflict with the variable condition on $\boldsymbol{\lambda}$-conversion: $\boldsymbol{x}$ is free in $\boldsymbol{\lambda y.}\mathbf{T}_{\boldsymbol{i}}(\boldsymbol{x},\boldsymbol{y})$, but then $\boldsymbol{\lambda P}$ lies in the scope of '$\exists \boldsymbol{x}$' in the matrix – or, more precisely, of the $\boldsymbol{\lambda}$ implicit in the notation '$\exists \boldsymbol{x}$'. So a renaming (97a) of the bound matrix variable (from $\boldsymbol{x}$ to brand-new $\boldsymbol{z}$)[22] needs to precede the conversion performed in (97b):

(97) a. $\neg(\exists \boldsymbol{x})[\mathbf{M}_{\boldsymbol{i}}(\boldsymbol{x}) \wedge \neg(\boldsymbol{\lambda P.}(\exists \boldsymbol{z})[\mathbf{F}_{\boldsymbol{i}}(\boldsymbol{z}) \wedge \boldsymbol{P}(\boldsymbol{z})])(\boldsymbol{\lambda y.}\mathbf{T}_{\boldsymbol{i}}(\boldsymbol{x},\boldsymbol{y}))]$

b. $\neg(\exists \boldsymbol{x})[\mathbf{M}_{\boldsymbol{i}}(\boldsymbol{x}) \wedge \neg(\exists \boldsymbol{z})[\mathbf{F}_{\boldsymbol{i}}(\boldsymbol{z}) \wedge (\boldsymbol{\lambda y.}\mathbf{T}_{\boldsymbol{i}}(\boldsymbol{x},\boldsymbol{y}))(\boldsymbol{z})]]$

Before going on, let us stop to emphasise that the variable condition was violated in the constellation $(\boldsymbol{\lambda P.}(\exists \boldsymbol{x})\ldots)(\boldsymbol{\lambda y.}\mathbf{T}_{\boldsymbol{i}}(\boldsymbol{x},\boldsymbol{y}))$ in(96) because $\boldsymbol{x}$ is free in the argument – notwithstanding the fact that $\boldsymbol{x}$ is bound by the outermost existential quantifier. In this case the side condition sees to it that an intervening binder – the innermost existential quantifier – destroys the binding relation between the $\boldsymbol{x}$ in the argument and its proper binder – the outermost existential quantifier. The latter binding relation turns on the fact that $\boldsymbol{x}$ is 'locally' free, i.e. free in the argument – which is precisely the criterion that is relevant for the side condition on $\boldsymbol{\lambda}$-conversion.

A final conversion now reduces (97b) to (98), which is not further reducible; this time the variable condition is obviously met, since the only variable occurring in the argument of the $\boldsymbol{\lambda}$-term is $\boldsymbol{z}$, which we have chosen so as to not appear anywhere in the whole formula, let alone in a binding prefix:

(98) $\neg(\exists \boldsymbol{x})[\mathbf{M}_{\boldsymbol{i}}(\boldsymbol{x}) \wedge \neg(\exists \boldsymbol{z})[\mathbf{F}_{\boldsymbol{i}}(\boldsymbol{z}) \wedge \mathbf{T}_{\boldsymbol{i}}(\boldsymbol{x},\boldsymbol{z})]]$

It is no accident that all reductions of the translation (92) of (91) considered so far come down to the same 'irreducible' formula (98). Rather, it is a consequence of a general property of type logic, known as *strong normalization*.

[22] The attentive reader will realise that we could have renamed $\boldsymbol{x}$ into $\boldsymbol{y}$ without getting into conflict with the side condition on $\boldsymbol{\lambda}$-conversion; in fact, this choice would have turned the next reduction step into an arguably simpler *eigen*-reduction. However, we decided to blindly stick to the more general safe strategy of always picking a 'fresh' variable.

According to it, (i) any type-logical formula can be reduced by a sequence of $\boldsymbol{\lambda}$-conversions and bound renamings to a so-called *normal form*; and, (ii) any two normal forms of a given formula are *alphabetic variants* of one another. Here a normal form is one that does not contain any sub-formula of the form '$(\boldsymbol{\lambda x.\alpha})(\boldsymbol{\beta})$', to which $\boldsymbol{\lambda}$-conversion may be applied (if need be after renaming); and alphabetic variants are formula that may be transformed into one another by a sequence of bound renamings. The proof of strong normalization is rather complex and would lead astray.[23] The result as such is of interest to indirect interpretation in that the exact procedure in reducing type-logical formulae is immaterial: as long as one does not miss any potentially relevant constellations of the form '$(\boldsymbol{\lambda x.\alpha})(\boldsymbol{\beta})$' and is prepared to rename bound variables whenever necessary, all paths lead to Rome – that is to say: the normal form.

Given strong normalization, we may rest assured that the decision to reduce (92) by starting with (93a) rather than (93b) does not bear on the end result (98). Indeed, (93b) contains two pertinent constellations – respectively starting with the $\boldsymbol{\lambda Q}$ in the first line and the $\boldsymbol{\lambda} P$ in the second line – which can be reduced in any order, similarly to the alternative paths leading from (94) to (96). In either case the result will be:

(99) $\quad (\boldsymbol{\lambda P.}\neg(\exists \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x})\wedge\neg \boldsymbol{P}(\boldsymbol{x})])(\boldsymbol{\lambda x.}(\exists \boldsymbol{z})[\mathbf{F}_i(\boldsymbol{z})\wedge(\boldsymbol{\lambda y.}\mathbf{T}_i(\boldsymbol{z},\boldsymbol{y}))(\boldsymbol{x})])$

Note that the elimination of the $\boldsymbol{\lambda P}$ involved renaming the bound $\boldsymbol{x}$ to $\boldsymbol{z}$ – just as in the parallel reduction (97). (99) again offers two options for $\boldsymbol{\lambda}$-reduction: getting rid of the $\boldsymbol{\lambda P}$ in the main functor or eliminating the $\boldsymbol{\lambda y}$ in the main argument. And again, the two reductions can be carried out in either order and lead to the same result:

(100) $\quad \neg(\exists \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x}) \wedge \neg(\boldsymbol{\lambda x.}(\exists \boldsymbol{z})[\mathbf{F}_i(\boldsymbol{z}) \wedge \mathbf{T}_i(\boldsymbol{z},\boldsymbol{x})])(\boldsymbol{x})]$

As the reader may check, in both conversions leading to (100), the side condition on the variables were fulfilled, saving us from any bound renaming. And, clearly, (100) now *eigen*-reduces to the above normal form (98) – just as the strong normalisation theorem would have it.

One detail concerning the renaming of bound variables is worth pointing out, before finally moving on to the indirect interpretation of intensional constructions. Let us therefore take a closer look at the above reduction from (97a) to (97b):

(101) $\quad \neg(\exists \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x})\wedge\neg(\boldsymbol{\lambda P.}(\exists \boldsymbol{x})[\mathbf{F}_i(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})])\ (\boldsymbol{\lambda y.}\mathbf{T}_i(\boldsymbol{x},\boldsymbol{y}))]$[=(96)]
$\equiv \quad \neg(\exists \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x})\wedge\neg\underline{(\boldsymbol{\lambda P.}(\exists \boldsymbol{z})[\mathbf{F}_i(\boldsymbol{z}) \wedge \boldsymbol{P}(\boldsymbol{z})])}\ \underline{(\boldsymbol{\lambda y.}\mathbf{T}_i(\boldsymbol{x},\boldsymbol{y}))}]$[=(97a)]

[23] It can be found in pertinent textbooks like *Introduction to combinators and lambda-calculus* (1986) by J. Roger Hindley and Jonathan Paul Seldin. The uniqueness (ii) of normal forms (up to alphabetic variation) is a version of the Church-Rosser Theorem mentioned in fn. 19 of Section 3.4 above.

$$\equiv \quad \neg(\exists \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x}) \wedge \neg(\exists \boldsymbol{z})[\mathbf{F}_i(\boldsymbol{z}) \wedge \mathbf{T}_i(\boldsymbol{x}, \boldsymbol{z})]] \qquad [=(97b)]$$

To prepare the $\boldsymbol{\lambda}$-conversion at the position indicated by underlining, we performed a bound renaming in the functor. As it turns out, we could just as well have proceeded like this:

$$\begin{array}{ll} (102) & \neg(\underline{\exists \boldsymbol{x}})[\mathbf{M}_i(\underline{\boldsymbol{x}}) \wedge \neg(\boldsymbol{\lambda P.}(\exists \boldsymbol{x})[\mathbf{F}_i(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})])\ (\boldsymbol{\lambda y.}\mathbf{T}_i(\underline{\boldsymbol{x}}, \boldsymbol{y}))] [=(96)] \\ \equiv & \neg(\underline{\exists \boldsymbol{z}})[\mathbf{M}_i(\underline{\boldsymbol{z}}) \wedge \neg(\boldsymbol{\lambda P.}(\exists \boldsymbol{x})[\mathbf{F}_i(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})])\ (\boldsymbol{\lambda y.}\mathbf{T}_i(\underline{\boldsymbol{z}}, \boldsymbol{y}))] \\ \equiv & \neg(\exists \boldsymbol{z})[\mathbf{M}_i(\boldsymbol{z}) \wedge \neg(\exists \boldsymbol{x})[\mathbf{F}_i(\boldsymbol{x}) \wedge \mathbf{T}_i(\boldsymbol{z}, \boldsymbol{x})]] \end{array}$$

In (102), too, the first transition involves renaming a bound variable: the binding of the variable $\boldsymbol{x}$, marked in the first line, has been replaced by a $\boldsymbol{z}$-binding; since $\boldsymbol{z}$ does not occur anywhere in the original formula (96), the conditions for applying the principle (64) of bound renaming are certainly met. And with this transition the conditions for applying $\boldsymbol{\lambda}$-conversion are again met; for the free variable $\boldsymbol{x}$ in the argument (= the $\boldsymbol{\beta}$-part, as it were) has disappeared by renaming, and the variable $\boldsymbol{z}$ taking its place has been chosen so as to not occur in the functor (= the $\boldsymbol{\alpha}$-part). In this respect, (101) and (102) are fully parallel; and the results are merely alphabetic variants of each other. However, there is a crucial difference between the two ways of proceeding, viz., that in (101) renaming applied in the $\boldsymbol{\alpha}$-part, whereas in (102) it was performed in the $\boldsymbol{\beta}$-part. Both strategies are fully legitimate *in this case*. Nevertheless it should be noted that the second strategy, followed in (102), only works if the variable in question – in this case: $\boldsymbol{x}$ – does get bound in the overall formula – in this case: by the existential quantifier '$\boldsymbol{(\exists x)}$'. If, however, the variable remains free throughout the whole formula, the corresponding renaming must not be performed – for only bound variables are to be renamed.[24] On the other hand, the strategy of bound renaming in the $\boldsymbol{\alpha}$-part, as in (101), can always be applied, for a simple reason: a violation of the variable condition of $\boldsymbol{\lambda}$-conversion is always the result of a conflict between a free variable in the $\boldsymbol{\beta}$-part and a binding in the $\boldsymbol{\alpha}$-part – which gets resolved by renaming the selfsame binding. To see how the strategy of renaming the trouble-making variable in the $\boldsymbol{\beta}$-part can go wrong, one may consider the sub-formula (103a) on which the above reduction (101) had been performed: renaming $\boldsymbol{x}$ in the $\boldsymbol{\beta}$-part before converting, as in (103b), would have led to the open formula (103c), which is clearly not equivalent to the starting point (103a) – since, unlike the latter, it depends on the value assigned to $\boldsymbol{z}$:

$$\begin{array}{lll} (103) & \text{a.} & (\boldsymbol{\lambda P.}(\exists \boldsymbol{x})[\mathbf{F}_i(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})])(\boldsymbol{\lambda y.}\mathbf{T}_i(\boldsymbol{x}, \boldsymbol{y})) \\ & \text{b.} & (\boldsymbol{\lambda P.}(\exists \boldsymbol{x})[\mathbf{F}_i(\boldsymbol{x}) \wedge \boldsymbol{P}(\boldsymbol{x})])(\boldsymbol{\lambda y.}\mathbf{T}_i(\underline{\boldsymbol{z}}, \boldsymbol{y})) \\ & \text{c.} & (\boldsymbol{\lambda P.}(\exists \boldsymbol{x})[\mathbf{F}_i(\boldsymbol{x}) \wedge (\boldsymbol{\lambda y.}\mathbf{T}_i(\underline{\boldsymbol{z}}, \boldsymbol{y}))(\boldsymbol{x})]) \\ & \equiv & (\boldsymbol{\lambda P.}(\exists \boldsymbol{x})[\mathbf{F}_i(\boldsymbol{x}) \wedge \mathbf{T}_i(\underline{\boldsymbol{z}}, \boldsymbol{x}))]) \end{array}$$

[24] This is not an arbitrary restriction but a result of the interpretation of type logic given in Section 5.3: in general, substitution of free variables does not preserve denotation.

To minimize the risk of breaking the denotational equivalence chain, it is thus strongly advised that in preparation of a $\boldsymbol{\lambda}$-conversion, *renamings of bound variables should always be restricted to the* $\boldsymbol{\alpha}$*-part.*

5.6 Simplifying notation

The constellation $\neg(\exists \boldsymbol{x})[\varphi \wedge \neg\psi]$ that we came across in the translation of (91) is so frequent that it deserves a notation of its own. On the whole, the formula expresses that no individual satisfies φ without satisfying ψ too – in other words, that everything that satisfies φ, also satisfies ψ.[25] In predicate logic this state of affairs is usually expressed by a combination of two logical constants. For one thing, the *universal quantifier* defined as an abbreviation in (104), expresses that a formula is satisfied by every individual. Like the existential quantifier, the universal quantifier may be construed as a formula of type $\boldsymbol{(et)t}$:

(104) a. $\forall$ is (an abbreviation for) the formula:
$(\boldsymbol{\lambda P^{et}.\neg(\exists x^{e})\neg P(x)})$
of type $\boldsymbol{(et)t}$.

b. If φ is a formula of type $\boldsymbol{t}$ and $\boldsymbol{x}$ is a variable of type $\boldsymbol{e}$, then $(\forall \boldsymbol{x})\varphi$ is (an abbreviation for) the formula:
$\forall((\boldsymbol{\lambda x.}\varphi))$.

As can be easily checked (in an exercise), (104) implies, for any assignment g:

(105) $\|(\forall \boldsymbol{x})\varphi\|^{g} = 1$ iff for any individual u the following holds:
$\|\varphi\|^{g[\boldsymbol{x}/u]} = 1$.

In predicate logic it is customary to read '$(\forall \boldsymbol{x})\ldots$' as 'for all x the following holds: …' or 'for every x the following holds: …', which seems justified by (105). As in the case of disjunction, we could have introuded $\forall$ as a logical constant (of type $\boldsymbol{(et)t}$) instead of an abbreviation; (105) would then have been a direct consequence of the interpretation of this constant. In any case, the universal quantifier can now be used to rewrite the constellation under scrutiny as follows:

(106) $\neg(\exists \boldsymbol{x})[\varphi \wedge \neg\psi]$
$\equiv$ $\neg(\exists \boldsymbol{x})\neg\neg[\varphi \wedge \neg\psi]$
$\equiv$ $(\forall \boldsymbol{x})\neg[\varphi \wedge \neg\psi]$

[25] We borrow the term *satisfaction* from predicate logic, where it is given a precise definition along the following lines: an individual u satisfies a formula φ *at position* $\boldsymbol{x}$ *relatively to* an assignment g iff φ is true if $\boldsymbol{x}$ denotes u – i.e., if φ denotes 1 given the modified assignment $g[\boldsymbol{x}/u]$.

The first transition in (106) is based on the Law of Double Negation (73a) from propositional logic.

There is a also a common abbreviation for the remaining constellation in (106):

(107) If φ and ψ are formulae of type $\boldsymbol{t}$, then
$[\varphi \rightarrow \psi]$
is (an abbreviation for) the formula:
$\neg[\varphi \wedge \neg\psi]$.

(107) immediately entails:

(108) $[\varphi \rightarrow \psi]$
$=$ $\neg[\varphi \wedge \neg\psi]$
$\equiv$ $[\neg\varphi \vee \psi]$

Apart from the Law of Double Negation, (108) only makes use of the notational introduction (75b) of disjunction $\vee$. And again, we could, of course, have introduced $\rightarrow$ as a separate logical constant (of type $\boldsymbol{t(tt)}$).

According to (108), a formula $[\varphi \rightarrow \psi]$ expresses that either φ is true or else – i.e. otherwise – ψ is false. *Up to a point*, this observation justifies the paraphrase 'if φ, then ψ', which is common in propositional and predicate logic. The arrow $\rightarrow$ and the truth table that goes with it also run by the name of *material implication*. However, as a semantic account of conditionals – as in **Wenn . . . , [dann] . . .** [$\approx$ If . . . , [then] . . .] – an interpretation along the lines of the constellation defined in (107) turns out to be insufficient, as we will come to see in Chapter 10.

Given (104) and (107), formulae of the form $\neg(\exists\boldsymbol{x})[\varphi \wedge \neg\psi]$ can be abbreviated as $(\forall\boldsymbol{x})[\varphi \rightarrow \psi]$. As the above translation of (91) shows, such formulae correspond to sentences of the form [[**Jed-** *N*] *VP*]. It is worth noticing that the symbol $\forall$ from predicate logic does not correspond to the determiner **jed-** [$\approx$ every]. For as we have seen in Chapter 3, the latter (or rather, its extension) establishes a relation between two sets (or rather their characteristic functions) – the relation of subset-hood. However, the universal quantifier, makes a statement about a single set – namely that every individual is an element of it. The difference between universal quantifier and determiner is thus reflected in their types: while the extension of **jed-** is of type $\boldsymbol{(et)((et)t)}$, the universal quantifier is a formula of type $\boldsymbol{(et)t}$. The extension of **jed-** is thus *binary* in that it takes *two* predicate extensions to obtain a truth value; the universal quantifier, on the other hand, is *unary*, in that it yields a truth value when given *one* predicate extension as its argument. Still, the above considerations show that the subset relation expressed

by the determiner can be expressed by the unary universal quantifier if the two predicate extensions to be related are combined with the help of material implication:

$$
\begin{array}{ll}
(109) & [\![\textbf{Jed-}\ N\ VP]\!]^s \\
= & \|\,|\textbf{Jed-}\ N\ VP|\,\|^s \\
= & \|(\forall \boldsymbol{x}^{\boldsymbol{e}})[|N|(\boldsymbol{x}) \rightarrow |VP|(\boldsymbol{x})]\|^s \\
= & \|\forall\|(\|(\boldsymbol{\lambda x}.[|N|(\boldsymbol{x}) \rightarrow |VP|(\boldsymbol{x})])\|^s) \\
= & \|\forall\|(\|(\boldsymbol{\lambda x}.[\neg|N|(\boldsymbol{x}) \vee |VP|(\boldsymbol{x})])\|^s) \\
= & \|\forall\|(\lambda x. \vdash [\![N]\!]^s(x) = 0 \text{ or } [\![VP]\!]^s(x) = 1 \dashv) \\
= & [\lambda P.\lambda Q.\|\forall\|(\underline{\lambda x. \vdash P(x) = 0 \text{ or } Q(x) = 1 \dashv})](([\![N]\!]^s)([\![VP]\!]^s)
\end{array}
$$

The first step of the transition makes use of the notational convention introduced in Section 5.4 to the effect that '$\|\boldsymbol{\alpha}\|^s$' is short for '$\|\boldsymbol{\alpha}\|^g$' whenever $Fr(\boldsymbol{\alpha}) = \{\boldsymbol{i}\}$ and $g(\boldsymbol{i}) = s$. The other steps will be reconstructed in an exercise. The final reformulation in (109) shows that the (subset-hood) relation the determiner **jed-** establishes between two sets can be construed as a property of the (underlined) combination of them, viz. the union of the second set with the complement of the first one (again identifying sets with their characteristic functions.

The *reducibility* of a binary determiner extension to a unary operation can also be observed in the case of other determiners. We thus have, in analogy to (109):

$$
\begin{array}{ll}
(110) & [\![\textbf{Ein-}_{indef}\ N\ VP]\!]^s \\
\ldots & \ldots \\
= & [\lambda P.\lambda Q.\|\exists\|(\lambda x. \vdash P(x) = 1 \text{ and } Q(x) = 1 \dashv)](([\![N]\!]^s)([\![VP]\!]^s)
\end{array}
$$

The binary relation (of *overlap*) expressed by the indefinite article thus turns out to be reducible to a combination of the two predicate extensions; however, this combination is different from the one used in the reduction (109) of $[\![\textbf{jed-}]\!]^s$ – viz. intersection. Yet not always can a corresponding reduction be given. As a case in point, the extension of **die meisten** cannot be rewritten as an operation on a combination of its two arguments. There is thus something anecdotal about the reducibility of $[\![\textbf{jed-}]\!]^s$, $[\![\textbf{ein-}_{indef}]\!]^s$ etc.[26] In view of this observation, many semanticists have dropped the existential and universal quantifiers of predicate logic for a more general and transparent notation, writing $(\forall \boldsymbol{x}\boldsymbol{:}\boldsymbol{\varphi})(\boldsymbol{\psi})$, $(\exists \boldsymbol{x}\boldsymbol{:}\boldsymbol{\varphi})(\boldsymbol{\psi})$, and $(\textbf{MOST}\boldsymbol{x}\boldsymbol{:}\boldsymbol{\varphi})(\boldsymbol{\psi})$, in *lieu* of $(\forall \boldsymbol{x})[\boldsymbol{\varphi} \rightarrow \boldsymbol{\psi}]$, $(\exists \boldsymbol{x})[\boldsymbol{\varphi} \wedge \boldsymbol{\psi}]$, and $\textbf{MOST}(\boldsymbol{\lambda x}.\boldsymbol{\varphi})(\boldsymbol{\lambda x}.\boldsymbol{\psi})$, respectively. Though there is certainly something to this notational innovation, we will

[26] It appears, though, that this reducibility was a decisive factor in the development of predicate logic. – The irreducibility of **die meisten** is a consequence of a theorem due to the US semanticist Ed Keenan, who proved it in his paper *Natural Language, Sortal Reducibility and Generalized Quantifiers* (1993).

stick to the traditional variant in these notes.

5.7 Intensionality

We end our indirect reformulation of the semantic analyses developed in the first chapters with the Hintikka semantics of attitude reports, at the same time taking the opportunity to extend the range of intensional constructions analyzed.

As already established in (2), the extension type of an attitude verb is ***(st)(et)***. When such a verb gets combined with a complement clause, the extension of the ensuing predicate is obtained by applying the verb *extension* to the *intension* of the complement. To capture this meaning combination within indirect interpretation, one first needs to find a way to turn a formula denoting the extension of a linguistic expression into one that denotes its intension; so far, the translation of the embedded sentence is of type $\boldsymbol{t}$ and thus denotes its truth value. At this point, it pays that the translations make explicit reference to the situation at hand – and that this reference is made by way of a variable. For the intension of an expression A is that function that assigns to every situation s in Logical Space its extension in s:

(111) $\quad [\![A]\!] = \lambda s.[\![A]\!]^s$

Since we assume that in the type-logical translation $|A|$ of A the situation at hand is denoted by the variable $\boldsymbol{i}$, the extension of A in $s(\in LS)$ can be obtained by assigning s to the variable $\boldsymbol{i}$:[27]

(112) $\quad [\![A]\!]^s = \|\,|A|\,\|^{g[\boldsymbol{i}/s]}$

Since (112) holds for arbitrary $s \in LS$,[28] (111) and (112) entail:

(113) $\quad [\![A]\!] = \lambda s.\|\,|A|\,\|^{g[\boldsymbol{i}/s]}$

In view of the formulation (48) ot the interpretation of $\boldsymbol{\lambda}$-abstraction, $\boldsymbol{(\lambda i.}|A|\boldsymbol{)}$ now emerges as the translation of the intension of A:

(114) $\quad \lambda s.\|\,|A|\,\|^{g[\boldsymbol{i}/s]} = \|\boldsymbol{(\lambda i.}|A|\boldsymbol{)}\|^g$

We thus obtain:[29]

[27] We are also assuming that the other variables do not play any rôle. In the translations considered so far, apart from $\boldsymbol{i}$, all variables are bound anyway; given the Coincidence Lemma, their denotation thus does not depend on the assignment.

[28] This requirement is indispensable to the inference from (111) and (112) to (113); for the 's' in (112) also stands for arbitrary situations. The point becomes clearer by reformulating (112) and (113) with different (meta-linguistic) situation variables – thus performing a bound renaming in the meta-language

[29] Cf. footnote 21 on the double bracketing in (115).

(115) *Indirect interpretation of* ***dass****-clauses as complementisers*
If P is a predicate consisting of an attitude verb V and a complement clause S, the following holds:
$|P| = |V|(\boldsymbol{(\lambda i.}|S|\boldsymbol{)})$.

It should be noted that it is not only essential for (115) that reference to the situation is made by a variable – otherwise we could not have abstracted from it; it is equally important that it is always the same variable, viz. $\boldsymbol{i}$ – otherwise we would not know which variable to bind to obtain the translation of the intension of the embedded clause.

To conclude the indirect formulation of Hintikka semantics, we still need to specify the translations of the lexical attitude verbs. To prepare the ground, we name the underlying perspectives *Dox*, *Epi*, *Bou*, ... by suitable type-logical constants **DOX**, **EPI**, **BOU**,.... As we have seen in Chapter 4, the (doxastic, epistemic, bouletic,...) perspective of an individual (of type $\boldsymbol{e}$) always depends on the given situation (of type $\boldsymbol{s}$) and consists of a set (of type $\boldsymbol{st}$) whose members are situations in Logical Space. Accordingly, the constants mentioned are formulae of type $\boldsymbol{e(s(st))}$, to be interpreted in the following way:

(116) a. $\|\mathbf{DOX}\| = \lambda x.\lambda s_0.\lambda s_1.\vdash s_1 \in Dox_{x,s_0} \dashv$
b. $\|\mathbf{EPI}\| = \lambda x.\lambda s_0.\lambda s_1.\vdash s_1 \in Epi_{x,s_0} \dashv$
c. $\|\mathbf{BOU}\| = \lambda x.\lambda s_0.\lambda s_1.\vdash s_1 \in Bou_{x,s_0} \dashv$

The constants interpreted in (116) can now be employed in the lexical translations of the attitude verbs:

(117) a. $|\mathbf{meint}| = \boldsymbol{(\lambda p^{st}.(\lambda x^{e}.(\forall j)[\mathbf{DOX}(x)(i)(j) \rightarrow p_j]))}$
b. $|\mathbf{weiß}| = \boldsymbol{(\lambda p^{st}.(\lambda x^{e}.(\forall j)[\mathbf{EPI}(x)(i)(j) \rightarrow p_j]))}$
c. $|\mathbf{will}| = \boldsymbol{(\lambda p^{st}.(\lambda x^{e}.(\forall j)[\mathbf{BOU}(x)(i)(j) \rightarrow p_j]))}$

As in (117), we will from now on use '$\boldsymbol{j}$' whenever we need an additional (bound) variable ($\neq \boldsymbol{i}$) of type $\boldsymbol{s}$; and we also write it as a subscript when used as an argument. Moreover, like before, $\boldsymbol{(\forall j)\varphi}$ is short for $\forall\boldsymbol{(\lambda j.\varphi)}$, where the universal quantifier $\forall$ abbreviates the formula $\boldsymbol{(\lambda p^{st}.\neg(\exists j)\neg p(j))}$ of type $\boldsymbol{(st)t}$ – like the existential quantifier used to define it. Strictly speaking, then, these universal and existential quantifiers are not the formulae introduced in the preceding sub-section, which had been of type $\boldsymbol{(et)t}$. However, in line with logical-semantic tradition, we refuse to make a notational difference between the two kinds of quantifiers.

To see the indirect version of the interpretation (115) and (117) of Hintikka semantics at work, we construct and reduce the type-logical translation of a simple attitude report:

(118) $|\textbf{Fritz meint, dass Eike einen Porsche besitzt}|$ by (78)
$= |\textbf{meint, dass Eike einen Porsche besitzt}|(|\textbf{Fritz}|)$ (115)[30]
$= |\textbf{meint}|(\boldsymbol{\lambda i}.|\textbf{Eike besitzt einen Porsche}|)(|\textbf{Fritz}|)$ by (74)
$= |\textbf{meint}|(\boldsymbol{\lambda i}.|\textbf{Eike besitzt einen Porsche}|)(\mathbf{f})$ by (117a)
$= (\boldsymbol{\lambda p^{st}}.(\boldsymbol{\lambda x^{e}}.(\forall j)[\mathbf{DOX}(x)(i)(j) \rightarrow p_j]))$
$(\boldsymbol{\lambda i}.|\textbf{Eike besitzt einen Porsche}|)(\mathbf{f})$ Exercise!
$\equiv (\boldsymbol{\lambda p}.(\boldsymbol{\lambda x}.(\forall j)[\mathbf{DOX}(x)(i)(j) \rightarrow p_j]))$
$(\boldsymbol{\lambda i}.(\exists y^{e})[\mathbf{P}_i(y) \wedge \mathbf{B}_i(\mathbf{e}, y)])(\mathbf{f})$ $\boldsymbol{\lambda}$-conversion (66)
$\equiv (\boldsymbol{\lambda x}.(\forall j)[\mathbf{DOX}(x)(i)(j) \rightarrow (\boldsymbol{\lambda i}.(\exists y)[\mathbf{P}_i(y) \wedge \mathbf{B}_i(\mathbf{e}, y)])(j)])(\mathbf{f})$
$\boldsymbol{\lambda}$-conversion (66)
$\equiv (\boldsymbol{\lambda x}.(\forall j)[\mathbf{DOX}(x)(i)(j) \rightarrow (\exists y)[\mathbf{P}_j(y) \wedge \mathbf{B}_j(\mathbf{e}, y)]])(\mathbf{f})$
$\boldsymbol{\lambda}$-conversion (66)
$\equiv (\forall j)[\mathbf{DOX}(\mathbf{f})(i)(j) \rightarrow (\exists y)[\mathbf{P}_j(y) \wedge \mathbf{B}_j(\mathbf{e}, y)]]$

Actually, (118) only captures one reading of the (surface) sentence under scrutiny. According to this reading, Fritz need not have a specific idea about Eike's possession. In particular, he need not have a particular car in mind that he takes Eike to own. In Chapter 7 we will encounter and analyse a further reading, according to which Fritz believes *of* a certain Porsche that *it* belongs to Eike.

The lexical analysis of (117c) not only covers finite complements; the use of **wollen** in (119), where it functions as a control verb and embeds infinitivals, can also be captured by this lexical entry (117c):

(119) **Fritz will gewinnen.**
[≈ Fritz wants to win.]

In order to interpret (119) in the framework of Hintikka semantics, we may (simplifyingly) assume that the infinitive may be paraphrased by a **dass**-clause whose (vacant) subject position is filled by the subject of the matrix clause:

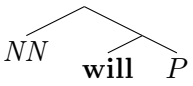
(120) a. *NN* **will** *P*
b. ≈ *NN* **will, dass** *NN P*

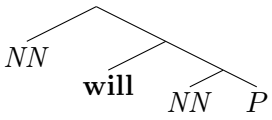
In (120), '*NN*' and '*V*' respectively stand for a proper name and a predicate; and we have neglected the difference between finite predicates and infinitves.[31] The synonmy effect postulated in (120) – this is how the quasi-equation ought to be read – can be achieved relatively easily on the basis of the lexical interpretation (117c) of **wollen**, under the assumption – and against prevalent syntactic ideas – that the embedded infinitive is underly-

[30] We continue to assume that the embedded **dass**-clause is reduced to its main clause variant.

[31] Basically this means that we have chosen to ignore the temporal reference expressed by tense morphology.

ingly subject-less and thus corresponds to a predicate:

(121) a. 

≈ b. 

According to (121), we should have:

(122)	$\lvert NN\ \textbf{will}\ P\rvert$	in view of (121)
≈	$\lvert NN\ \textbf{will, dass}\ NN\ P\rvert$	by (78) & (115)
≡	$\underline{\lvert\textbf{will}\rvert(\boldsymbol{\lambda i}.\lvert P\rvert(\lvert NN\rvert))}(\lvert NN\rvert)$	

The underlined part obviously corresponds to the translation of the predicate of (120b): its denotation characterizes the set of individuals who want that *NN* (= the bearer of the name '*NN*') will win. Yet despite the (assumed) synonymy between the two attitude reports, the sub-formula highlighted in (122) does not correspond to the predicate of (120a); the extension of the latter comprises those individuals that would like to win themselves: in (120a), the proposition functioning as the argument of the modal verb **will** depends on the subject. The following reformulation of the translation of (119) accounts for this dependence:

(123)	$\lvert NN\ \textbf{will}\ P\rvert$	cf. (122)
≈	$\lvert\textbf{will}\rvert(\boldsymbol{\lambda i}.\lvert P\rvert(\lvert NN\rvert))(\lvert NN\rvert)$	$\boldsymbol{\lambda}$-conversion (66)
≡	$\underline{(\boldsymbol{\lambda x^e}.\lvert\textbf{will}\rvert(\boldsymbol{\lambda i}.\lvert P\rvert(\boldsymbol{x}))(\boldsymbol{x}))}(\lvert NN\rvert)$	

This time the underlined part corresponds to the predicate of (119): its denotation characterizes the set of individuals who want that they themselves win. That the transition is correct follows from the fact that the translations of proper names are always constants and thus do not contain any variables that might be bound during conversion.

From (123) we immediately get an interpretation of infinitival embedding according to which the extension of the control verb is of type ***(st)(et)***:

(124) *Indirect interpretation of infinitival embedding under (subject) control verbs*
If *VP* is a predicate consisting of an attitude verb *V* and an infinitival predicate *P* as its object, the following holds:
$$\lvert VP\rvert = (\boldsymbol{\lambda x^e}.\lvert V\rvert(\boldsymbol{\lambda i}.\lvert P\rvert(\boldsymbol{x}))(\boldsymbol{x}))$$

The above interpretation only works for such verbs that equate (or *control*, in the syntactician's parlance) the implicit subject of their complement ('*PRO*') with their subject, in the sense of the (near) synonymy (120). Among these are (certain uses of) German modal verbs but also ordinary verbs embedding infinitives under **zu** [≈ to]: **hoffen** [≈ hope], **versprechen** [≈ promise],

versuchen [≈ try]. The latter example shows that some of these verbs do not even embed finite clauses, but can still be analyzed along the lines of (124). However, not every infinitival embedding can be interpreted as in (124). On the one hand, some verbs (**anweisen** [≈ instruct], **empfehlen** [≈ recommend], **raten** [≈ advise],...) equate the infinitival subject with an object of the embedded verb; in this case one may give an interpretation of the object-control construction that is analogous to (124); we abstain from doing so for reasons of time and space. On the other hand, a handful of verbs, intuitively speaking, treat their subject as part of the infinitival, rather than equating it with its subject. Among these so-called *raising verbs* are again (certain readings of) German modal verbs plus certain uses of the full verbs **scheinen** [≈ appear] and **drohen** [≈ threaten]. That such verbs cannot be captured by (124) is best seen when considering a quantifying subject:

(125) **Die meisten Gäste scheinen da zu sein.**
[≈ Most visitors appear to be here.]

(125) is true of a situation in which the speaker has some (not further specified) evidence that the majority of visitors of a certain (nor further specified) event have already arrived at the (nor further specified) venue. This evidence may, e.g., consist in the state of a coat stand: the fact that five coats are hanging on the visitors' coat rack indicates that already five of the altogether seven invited guests have arrived. A speaker who relies on this evidence does not necessarily have been able to match even a single garment to its wearer. He need not even be able to utter a qualified conjecture about a single visitor and the question of whether she has arrived. In other words: even if (125) is true, for none of the expected guests *NN* need the following sentence be true:

(126) ***NN* scheint da zu sein.**
[≈ *NN* appears to be here.]

This, however, means that **scheinen** cannot be an attitude verb (of extension type ***(st)(et)***) to which (124) applies whenever it embeds an infinitival. For the translation (124) of predicates '*V* **zu** *P*' combines with the quantifying subject **die meisten Gäste** so that, following the familiar account (88) of quantification, the truth of the resulting quantified sentence depends on how many corresponding predications are true; accordingly (125) would be true if there were more true sentences of the form (126) (where each '*NN*' stands for a guest) than false ones.[32] The example showed that things are different: irrespective of the truth of (125), the predications (126) could all be false.

The reason why raising verbs like **scheinen** are not construed along the lines of (124) is that, though they do express propositional attitudes (of sorts),

[32]We are tacitly assuming that each visitor has exactly one name!

they do not ascribe them to their subject (or, more precisely, its extension). In the case of **scheinen** the rôle of the missing subject can be expressed by an '*Experiencer*'-dative: **ihm scheint, dass ....** This case-anomaly notwithstanding, this use of **scheinen** can obviously be captured within the Hintikka semantics framework. We will not go into the details of this construction here, but take it that, as in the other cases of attitudes analyzed in (116), there is an underlying alternative-relation of type $\boldsymbol{e(s(st))}$:

(127) $\|\mathbf{EVI}\| = \lambda x.\lambda s_0.\lambda s_1.\ \vdash$ in s_0, is s_1 an evidential alternative for x $\dashv$,

where the *evidential* alternatives (for an individual x in a given situation s_0) are those possible situations that x cannot exclude to be in, given the evidence available to x in s_0. We are deliberately leaving open what counts as evidence here – be it perception, previous knowledge, hearsay, etc.

The Hinitkka-style attitude relation in (127) also underlies the *impersonal* use of **scheinen** [≈ appear] – as in: **Es scheint, dass ...** [≈ It appears that ...], where the rôle of the attitude bearer is implicitly played by the speaker, or a group (s)he belongs to. Since we are taking no account of context dependence – and particularly reference to the speaker – for the time being (more specifically until Chapter 8, we will simplify matters and talk of the evidential alternatives 'in a situation', for which we reserve a constant $\mathbf{EVI}^*$ of type $\boldsymbol{s(st)}$:[33]

(128) $\|\mathbf{EVI}^*\| = \lambda s_0.\lambda s_1.\ \vdash s_1$ is an evidential alternative in s_0 $\dashv$

On the basis of $\mathbf{EVI}^*$, the impersonal use of **scheinen** can be analysed like an attitude verb, except that it lacks a (personal) subject. Here is a case in point:

(129) $|\mathbf{Es\ scheint,\ dass\ die\ meisten\ Gäste\ da\ sind}|$
$\equiv$ $(\forall \boldsymbol{j})[\mathbf{EVI}^*_{\boldsymbol{i}}(\boldsymbol{j}) \rightarrow \mathbf{MOST}(\boldsymbol{G_j})(\boldsymbol{D_j})]$
$\equiv$ $|\mathbf{schein\text{-}}|(\boldsymbol{\lambda i}.|\mathbf{die\ meisten\ Gäste\ sind\ da}|)$

We leave out the – hardly exciting – details of this analysis. What matters is merely that in (129), we have assumed the following lexical interpretation of impersonal **scheinen**:

(130) $|\mathbf{schein\text{-}}| = (\boldsymbol{\lambda p^{st}}.(\forall \boldsymbol{j})[\mathbf{EVI}^*_{\boldsymbol{i}}(\boldsymbol{j}) \rightarrow \boldsymbol{p_j}])$

The fact that (125) seems to express the same as the impersonal variant analysed in (129), suggests that (130) also underlies the use of **scheinen**

[33] Objects of this type can obviously be construed as relations between points in Logical Space. In modal logic (where Hintikka semantics has its origins), they are known as *accessibility relations*.

as a raising verb. The (indirect) interpretation of (125) would thus have to account for the following sequence of equivalences:

(131) $|\textbf{die meisten Gäste scheinen da zu sein}|$ cf. (129)
$\equiv$ $|\textbf{scheint}|(\boldsymbol{\lambda i}.|\textbf{die meisten Gäste sind da}|)$ by (88)
$\equiv$ $|\textbf{scheint}|(\boldsymbol{\lambda i}.|\textbf{die meisten Gäste}|(|\textbf{sind da}|))$ by (74)
$\equiv$ $|\textbf{scheint}|(\boldsymbol{\lambda i}.\underline{\textbf{MOST}(\boldsymbol{G_i})}(\boldsymbol{D_i}))$

In order to achieve this result in a compositional fashion, the predicate **scheinen da zu sein** needs to be combined with the subject **die meisten Gäste** so that the latter can take the underlined position in the argument of $|\textbf{scheint}|$. The translation of the predicate would thus have to look like the last line of (131) – minus the underlined part. The kind of subtraction needed here can of course again be performed by $\boldsymbol{\lambda}$-abstraction. However, care needs to be taken to avoid a complication with the situation variable $\boldsymbol{i}$. For the following fomula is *not* equivalent to the last line in (131):

(132) $(\boldsymbol{\lambda X^{(et)t}}.|\textbf{scheint}|(\boldsymbol{\lambda i}.\boldsymbol{X}(\boldsymbol{D_i})))(\textbf{MOST}(\boldsymbol{G_i}))$

We skip the proof of non-equivalence; but take notice of the fact that the transition from (131) to (132) is not supported by the law (66) of $\boldsymbol{\lambda}$-conversion: the argument $\textbf{MOST}(\boldsymbol{G_i})$ (= $|\textbf{die meisten Gäste}|$) contains the free variable $\boldsymbol{i}$ that gets bound in the substitution process. That (132) does not work as a decomposition of the translation of (125) also shows in the fact that the argument merely denotes the *extension* of the subject. According to (132), then, any co-extensional subject would lead to the same truth value – contrary to fact: if the visitors happen to be precisely the members of the rabbit breeder society, the seeming arrival of their majority does not necessarily constitute evidence for most visitors' arrival. (We leave it to the readers to construct a pertinent scenario in which extensional substitution of the subject fails.) So the contribution that the subject makes to the truth value of (125) is not confined to its extension, but rather lies in its intension. Accordingly, the formula in the final line of (131) needs to be dissected as follows:

(133) $|\textbf{scheint}|(\boldsymbol{\lambda i}.\textbf{MOST}(\boldsymbol{G_i})(\boldsymbol{D_i}))$
$\equiv$ $|\textbf{scheint}|(\boldsymbol{\lambda i}.\underline{(\boldsymbol{\lambda i}.\textbf{MOST}(\boldsymbol{G_i}))}(\boldsymbol{i})(\boldsymbol{D_i}))$ *eigen*-conversion (72)
$\equiv$ $(\boldsymbol{\lambda \wp^{s((et)t)}}.|\textbf{scheint}|(\boldsymbol{\lambda i}.\boldsymbol{\wp}(\boldsymbol{i})(\boldsymbol{D_i})))(\underline{\boldsymbol{\lambda i}.\textbf{MOST}(\boldsymbol{G_i})})$
$\boldsymbol{\lambda}$-conversion (66)

The sub-formula underlined in (133) corresponds to the intension of the subject. Thus the rest of the formula ought to correspond to the translation of the predicate, which in turn ought to be derivable by combining the translation (130) of the raising verb with the embedded infinitival. This is achieved by the following, by now pretty obvious, rule:

(134) *Indirect interpretation of infinitival embedding under raising verbs*
If VP is a predicate consisting of a raising verb V and an infinitival predicate P as its complement, the following holds:
$$|VP| = (\boldsymbol{\lambda}\wp^{\boldsymbol{s((et)t)}}\boldsymbol{.}|V|(\boldsymbol{\lambda i.}\wp(\boldsymbol{i})(|P|)))$$

(134) can be combined with the intension of the subject by functional application. However, this cannot be achieved by the quantificational rule (88); since predication does not work either in this case – the subject of (125) is a quantifier after all and not of type $\boldsymbol{e}$ –, we now obviously need a third way of construing the connection between subject and predicate:

(135) *Indirect interpretation of raised subjects*
If S is sentence whose subject is a quantifying noun phrase QN and whose predicate VP translates as a type-logical formula $|VP|$ of type $\boldsymbol{s((et)t))t}$, the following holds:
$$|S| = |VP|(\boldsymbol{\lambda i.}|QN|)$$

(135) is a new kind of rule in that it explicitly refers to the type of the translation of one of the constituents involved. This – methodologically innocuous – complication could be avoided if the pertinent predicates were syntactically marked (e.g., by a feature [± Raising]. The term 'raised subjects' reflects the intuition that the quantifying subjects behave as if they had been raised from the subordinate infinitival clauses. This intuition could also be directly modeled in syntactic terms, thus giving rise to an alternative treatment and interpretation of the construction to which we will get in Chapter 7 .[34]

(134) and (135) only cover raising verbs with quantifying subjects and thus cannot be immediately applied to sentences of the form (126). This gap will be closed in the next chapter, where we will encounter a method of reducing predication, quantification, and raising to a single construction.

What is special about the interpretation (135) of the raising verb construction is the fact that, (i) the subject position is occupied by a quantifier, whereas (ii) it is interpreted as an argument of the predicate extension. With respect to the first trait (i), the construction resembles quantification, whereas (ii) is reminiscent of predication. This combination of properties of the subject position of raising verbs, can also be found in the object position of so-called *opaque* verbs – a case in point being:

(136) **Fritz sucht ein**$_{indef}$ **Restaurant.**

[34]The above account of raising verbs was first mentioned, but not worked out in detail, in Richard Montague's paper *The Proper Treatment of Quantification in Ordinary English* (1973). Semantic composition mechanisms that explicitly refer to the (extension) types of the constituents involved have been made popular by Irene Heim and Angelika Kratzer through their 1998 text-book *Semantics in Generative Grammar*.

[≈ Fritz is looking for a restaurant.]

Just like the interpretation (88) of quantifying subjects failed on (125), in this case the interpretation (89) of quantifying objects leads to problems. For if (136) is true of a situation s, Fritz need not be looking for a particular restaurant in that situation; in other words, there need not be a restaurant called 'L' for which (137) holds:

(137) **Fritz sucht *L*.**
[≈ Fritz is looking for L.]

However, according to the rule of object quantifiers (89), (136) expresses that (137) is true for some restaurant L. But then (136) can even be true when there are no restaurants at all in the given situation. In order to still arrive at a compositional account of (136), let us first take a look at the following near-paraphrase:[35]

(138) **Fritz will ein$_{indef}$ Restaurant finden.**
[≈ Fritz wants to find a restaurant.]

The interpretive machinery developed so far immediately accounts for (138). As is readily seen, the rule (124) for control verbs leads to the following (reduced) type-logical translation of (138):

(139) $(\forall j)[\mathbf{BOU}(\mathbf{f})(i)(j) \rightarrow (\exists y^{e})[R_j(y) \wedge F_j(\mathbf{f}, y)]]$

Under the assumption that (136) is synonymous with (138), (139) can be used to arrive at a compositional interpretation of the sentence. The procedure is the same as in the case of **scheinen** in (131) and (133), i.e. we first isolate the contribution the object makes to the extension of the predicate:

(140)
$|\mathbf{sucht\ ein}_{indef}\ \mathbf{Restaurant}|$ by assumption
$\equiv$ $|\mathbf{will\ ein}_{indef}\ \mathbf{Restaurant\ finden}|$ by (89) & (124)
$\equiv$ $(\lambda x^{e}.|\mathbf{will}|(\lambda i.|\mathbf{ein\ Restaurant}|(\lambda y^{e}.|\mathbf{finden}|(x, y)))(x))$ by (76b) etc.
$\equiv$ $(\lambda x.|\mathbf{will}|(\lambda i.(\lambda P^{et}.(\exists y)[R_i(y) \wedge P(y)])(\lambda y.|\mathbf{finden}|(x, y)))(x))$ *eigen*-conversion (72)
$\equiv$ $(\lambda x.|\mathbf{will}|(\lambda i.\underline{(\lambda i.(\lambda P.(\exists y)[R_i(y) \wedge P(y)]))}(i)$
$(\lambda y.|\mathbf{finden}|(x, y)))(x))$ $\boldsymbol{\lambda}$-conversion (66)

[35] The idea to reduce opaque verbs to propositional attitudes by paraphrase goes back to the US-logician and philosopher Willard Van Orman Quine, who developed it in his paper *Quantifiers and Propositional Attitudes* (1956). The compositional account of opacity originates with Richard Montague, who first sketched it in his paper *On the Nature of Certain Philosophical Entities* (1968) and later elaborated it in *Universal Grammar* (1970) and *The Proper Treatment of Quantification in Ordinary English* (1973).

$\equiv \quad (\boldsymbol{\lambda}\wp^{s((et)t)}\boldsymbol{.}(\boldsymbol{\lambda x.}|\mathbf{will}|(\boldsymbol{\lambda i.}\wp(i)(\boldsymbol{\lambda y.}|\mathbf{finden}|(x,y)))(x)))$
$(\boldsymbol{\lambda i.}(\boldsymbol{\lambda P.}(\exists y)[R_i(y) \wedge P(y)]))$

The first transition in (140) is an immediate result of the translation rules mentioned (plus the usual assumptions about the lexical meanings involved). The second transition makes explicit reference to the translation of the object in order to bring to attention the free variable $\boldsymbol{i}$. In the third transition the intension of **a restaurant** is isolated as the contritbution to the extension; as in (133), this is done with the help of *eigen*-conversion. In this way the ground is prepared for the conversion to follow; otherwise the free $\boldsymbol{i}$ would have been bound – as in (132). Starting from the synonymy of the predicates in (136) and (138), the final transition brings out clearly that we have isolated the intension of the object. The rest of the formula, then, is the contribution of the predicate:

(141) $|\mathbf{sucht}|$
$= \quad (\boldsymbol{\lambda}\wp^{s((et)t)}\boldsymbol{.}(\boldsymbol{\lambda x}^{e}\boldsymbol{.}|\mathbf{will}|(\boldsymbol{\lambda i.}\wp(i)(\boldsymbol{\lambda y}^{e}\boldsymbol{.}|\mathbf{finden}|(x,y)))(x)))$
$\equiv \quad (\boldsymbol{\lambda}\wp\boldsymbol{.}(\boldsymbol{\lambda x.}(\forall j)[\mathbf{BOU}(x)(i)(j) \rightarrow \wp(j)(\boldsymbol{\lambda y.}\mathbf{F}_j(x,y))]))$

The lexical analysis in (141) obviously requires the following translation rule:

(142) *Indirect interpretation of the objects of opaque verbs*
If VP is a predicate consisting of an opaque verb V and a quantifying noun phrase QN, the following holds:
$|VP| = |V|(\boldsymbol{\lambda i.}|QN|)$

Analyses like (141), which explicitly refer to the meanings (or intensions) of other lexical expressions (like **will** [≈ want] and **finden** [≈ find]) are sometimes called *lexical decompositions*.

Just like, according to (135), (the extension of) the raising verb takes (the intension of) the quantifying subject as an argument, so (142) has (the extension of) the opaque verb operate on (the intension of) the object. Moreover (and despite the different lengths of the formulae involved), there is an interesting parallel between the lexical analysis (130) of the raising verb **scheinen** and the translation (141) of the opaque verb **suchen**: in both cases, the verb meaning is compositionally reduced to a propositional attitude, which is then interpreted in terms of Hintikka-style perspectives. In the case of the raising verb **scheinen** this was the relation expressed by the impersonal use of **scheinen**. In the case of the opaque verb **suchen** it was the bouletic perspective expressed by **wollen**. This may be no coincidence; from the so-called *propositionalist* stance, all intensional constructions can be paraphrased by clausal embeddings. On this view (which we cannot go into here), raising verbs and opaque verbs are only a special case. What

speaks in favor of propositionlism is that for practically all known opaque verbs, (more or less) plausible lexical decompositions along the lines of (141) can be found. As a case in point, the opaque verb **schulden** [≈ owe] can be analyzed by combining the meanings of **müssen** [≈ must] and **geben** [≈ give], as will be shown in an exercise.

As remarked above, Fritz need not be looking for a particular restaurant for the sentence (136) to be true. On the other hand, he may very well be looking for a particular restaurant, which will then guarantee the truth of (136). In this case we are dealing with the so-called *specific reading* (of the object). We will get to it in the next chapter but one. The analysis given in (142) only covers the non-specific reading.

5.8 Alternatives to two-sorted type logic

Apart from the two-sorted type logic used for indirect interpretation in this chapter, there are a number of other formal languages that are more or less suited for this purpose. Two of them will be introduced in this section. Before that we will, however, first address a variant of two-sorted type logic that is frequently found in the semantic literature.

5.8.1 Logical constants

In (33) in Section 5.3, we had introduced four 'logical' constants that are of supreme relevance for indirect interpretation in that they are employed for quite a few diverse purposes. Moreover, we had introduced further symbols ($\vee$,$\rightarrow$, and $\forall$) as abbreviations for formulae that could be formed from these logical constants – and which we could as well have treated as constants themselves. By treating them as abbreviations, we could show that they are *reducible* to the other logical constants – '$\wedge$', '$\exists$', '$\neg$', and '$=$': one could, in principle, do with the logical constants defined in (33) alone.

In principle, one can even do without the logical constants defined in (33). However, in order to do so, one needs to extend the syntax of type logic by one more construction, viz. equations between formulae of *any* types:

(143) *Syntax of (two-sorted functional) type logic with equations*
For all types a and b the following holds:

(Var) Variables of type a are formulae of type a.

(Con) Constants of type a are formulae of type a.

(App) If $\boldsymbol{\alpha}$ is a formula of some type (ab) and $\boldsymbol{\beta}$ is a formula of type a, then $\boldsymbol{\alpha(\beta)}$ is a formula of type b.

(Abs) If $\boldsymbol{x}$ is a formula of some type a and $\boldsymbol{\alpha}$ is a formula of some type b, then $\boldsymbol{(\lambda x.\alpha)}$ is a formula of type (ab).

(Id) If $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are formulae of the same type a, then $\boldsymbol{(\alpha = \beta)}$ is a formula of type $\boldsymbol{t}$.

Only clause *(Id)* is new; the other ones are as in (21). However, the version of type logic given in (143) also does without the logical constants defined in (33); for they are going to be construed as abbreviations.

It should be noted that the two formulae in equations need to be of the same type and that the resulting type is always $\boldsymbol{t}$; hence an equation between formulae of distinct types is not interpretable, and an equation between formulae of the same type always stands for a truth value. For the special case in which the type of the formulae equated is $\boldsymbol{e}$, we get equations that we could also have formed with the help of the logical constant '$=$' of type $\boldsymbol{e(et)}$, as defined in (33d); by analogy, we could have dropped clause *(Id)* in (143) and introduced a constant '$=_a$' for each type a.

The interpretation of type logic according to (143) proceeds as before – i.e. by the interpretive clauses given in Section 5.3: (29), (30), (35), and (46) (or, equivalently (47) or (48)), as well as by a further, obvious semantic rule for equations:

(144) *Interpretation of identity (Id)*
If g is an assignment and $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are formulae of some type a, the following holds:

$$\|\boldsymbol{\alpha} = \boldsymbol{\beta}\|^g = \vdash \|\boldsymbol{\alpha}\|^g = \|\boldsymbol{\beta}\|^g \dashv.$$

For the special case that the two equated formulae $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ are of type $\boldsymbol{t}$, we have: $\|\boldsymbol{\varphi} = \boldsymbol{\psi}\|^g = |1 - \|\boldsymbol{\varphi}\|^g + \|\boldsymbol{\psi}\|^g|$. In formal logic, this combination of truth values is known as 'material equivalence' and written '$[\boldsymbol{\varphi} \leftrightarrow \boldsymbol{\psi}]$'. It is readily seen that material equivalence is logically equivalent to the conjunction of material implications in both directions: $[\boldsymbol{\varphi} \leftrightarrow \boldsymbol{\psi}] \equiv [[\boldsymbol{\varphi} \rightarrow \boldsymbol{\psi}] \wedge [\boldsymbol{\psi} \rightarrow \boldsymbol{\varphi}]]$. Further truth tables can be defined with the help of identity, although sometimes in a somewhat roundabout way. We will only show this for ($\neg$) and conjunction ($\wedge$).[36] To prepare the ground for negation, we define an equation $\perp$ that always – i.e., given any assignment – denotes the truth value 0. $\perp$ says that (the characteristic function of) the set of all truth values $\{0, 1\}$ equals (the characteristic function of) the singleton set $\{1\}$ – which is, of course, wrong. The set of all truth values is the set of all truth values that are identical to themselves – $\{0, 1\} = \{u \in \{0, 1\} |\ u = u\}$ – and is therefore

[36] In fact, *arbitrary* truth tables can be defined in two-sorted type logic with identity; this follows from a fundamental theorem from propositional logic, the *functional completeness* of conjunktion and negation, which means that any combination of truth values can be reduced to a combination of $\wedge$ and $\neg$. A proof may be found on p. 38 of the class notes *Formale Grundlagen der Sprachphilosophie*: `https://user.uni-frankfurt.de/~tezimmer/Zimmermann/FGdSp.pdf`.

characterisable by the formula $\boldsymbol{\lambda}$-term $(\lambda u.(u = u))$, which corresponds to an expression of type logic with identity (between truth values). Moreover, $\{1\}$ is the set of truth values that are identical to 1: $\{u \in \{0,1\}|\ u = 1\}$ – which truth value in turn is the denotation of any equation of the form $\boldsymbol{(\alpha = \alpha)}$. Given these observations we arrive at the following definition:

(145) $\quad \bot := \boldsymbol{(\lambda u^t.(u = u)) = (\lambda u.(u = (u = u)))}$

The proof that, indeed, (for any g) $\|\bot\|^g = 0$, is left to the readership. On the basis of (145), negation can now be defined; for, given any truth value u, the statement that u is identical to 0, always has the opposite truth value of u:

(146) $\quad \neg := \boldsymbol{(\lambda u^t.(u = \bot))}$

Conjunction is more complicated. One way of expressing it in terms of the expressive means given in (143) makes use of a set-theoretic version of what is known as *Leibniz's Principle*; according to it, any objects x and y are identical to each other just in case x has the very same properties as y:[37]

(147) If U is a set and $\{u, v\} \in U$, the following holds:
$u = v$ iff for all subsets $X \subseteq U$: $u \in X$ iff $v \in X$.

The principle (147), which is valid in set theory, looks more complicated than it actually is. To begin with, only one direction is of any interest. For the fact that u is an element of the same sets as v if u and v are identical, is obvious anyhow: if $u \in X$, it immediate follows that, given $u = v$, $v \in X$ – and vice versa. However if – and now we are getting to the interesting direction – u and v are elements of the same subsets of X, we have, in particular: $v \in \{u\}$, for $\{u\}$ is a subset of U of which u is an element. But $\{u\}$ also contains no other elements, and we may therefore conclude that $u = v$, given that $v \in \{u\}$, by the assumption that v is a member of the same subsets as u.

The connection between the set-theoretic version (147) of Leibniz's Principle and conjunction emerges from the observation (already made in the first chapter) that the latter can be construed as a function from pairs of truth values to truth values – and thus as the characteristic function of a set of pairs of truth values. Since conjunction assigns the truth value 1 to a pair (u, v) of truth values just in case $u = v = 1$, it characterizes the singleton set $\{(1,1)\}$, i.e. the set of all objects that are identical to the pair $(1,1)$: $\{(u,v)|\ (u,v) = (1,1)\}$. According to (147), conjunction thus characterizes

[37] More precisely this is the *Principle of the Identity of Indiscernibles* [*principium identiatis indiscernibilium*], which is frequently attributed to Gottfried Wilhelm Leibniz (1646–1716), although it has been known since antiquity. The exact relation between this principle and its type-logical version is complex and would lead us astray.

the set of pairs (u, v) that are members of the same set of truth value pairs as $(1, 1)$:

(148) $\downarrow[\![\mathbf{und}]\!]^s$
$= \{(1,1)\}$
$= \{(u,v) |\ (u,v) = (1,1)\}$
$= \{(u,v) |$ for all sets X of pairs of truth values: $(u,v) \in X$ iff $(1,1) \in X\}$

In the type hierarchy, where sets of pairs (of truth values) are represented by sets assigning characteristic functions of sets (of truth values to truth values), (148) can be reformulated like this:

(149) $[\![\mathbf{und}]\!]^s = \lambda v.\lambda u.\vdash$ for all R of type $\boldsymbol{t(tt)}$: $R(v)(u) = R(1)(1) \dashv$

The condition to the right of the lambdas in (149) can now be construed as a combination of abstractions, applications, and identities and thus expressed in type logic with identity. To see this, we only have to express the locution 'for all R of type a: ...'. This is relatively simple; for a statement '...' is true of all objects of some type a just in case the set of those objects of type a that satisfy '...' coincides with the set of all objects of this type – i.e. with the set of self-identical objects of type a. We thus arrive at a (general) definition of the universal quantifier over objects of a type a:

(150) $\forall^a := (\boldsymbol{\lambda P^{at}.((\lambda x^a.P(x)) = (\lambda x.(x = x))))}$

By the so-called *duality* – the inter-definability – of universal and existential quantifier (with the help of negation), (150) allows for a general definition of the existential quantifier for arbitrary types:

(151) $\exists^a := (\boldsymbol{\lambda P^{at}.\neg\forall^a(\lambda x^a.\neg P(x)))}$

For the special case that $a = \boldsymbol{e}$, the formula in (151) is provably equivalent to the formula used in the interpretation (33b) of the existential quantifier as a logical constant. The same holds for the formula (150) and the version of the universal quantifier introduced in (104). For the case $a = \boldsymbol{t(tt)}$, (151) can finally be exploited to give a type-logical version of (149):[38]

[38] An alternative definition, due to Leon Henkin's *A theory of propositional types* (1963), makes use of the observation that the pair (1,1) is already characterised by the behaviour of the truth values on unary functions f of type $\boldsymbol{tt}$: $u = v = 1$ iff the f for which (i) $f(1) = 1$ are the same as the f for which (ii) $f(u) = v$. This is so because only two f satisfy (i), viz. the identical mapping id, and the function f_1 mapping every truth value to 1; but from $f_i(u) = v$ it follows that $u = v$, and from $id(u) = v$ it follows that $v = 1$ so that we may conclude that $u = v = 1$, as required. Hence the type-logical formula $(\boldsymbol{\lambda v^t.(\lambda u^t.(\forall f^{tt})((f(u) = v) = (f(\neg\bot) = \neg\bot))))}$ may be used to define conjunction; and it can even be reduced to $(\boldsymbol{\lambda v.(\lambda u.(\forall f)((f(u) = v) = f(\neg\bot))))}$, given the logical equivalence of $(\boldsymbol{\varphi = \neg\bot})$ and any $\boldsymbol{\varphi}$ of type $\boldsymbol{tt}$! – There is a further definition of conjunction due to Tarski, which however is too obscure to be mentioned

(152) $\wedge := (\boldsymbol{\lambda v^t.(\lambda u^t.(\forall^a R^{t(tt)})[R(v)(u) = R(\neg\bot)(\neg\bot)])})$

This finishes our demonstration that type logic with identity can do without the four logical constants. Remarkable though this fact is from a logical point of view, it is irrelevant to the business of indirect interpretation. On the other hand, the reductions in (145), (146), and (150)–(152) reflect a common feature of the logical constants that is also of semantic interest. For the fact that they can be defined by purely logical means – that is, without recourse to any constants – implies their universal applicability, irrespective of the thematic context. In logic, this topic neutrality is known as *invariance* – a term that we have already come across in connection with determiner extensions and that we will now generalize. The basic idea is that the denotations of logical constants are functions that only relate to the structural aspects of their arguments. It is a main characteristic of these structural aspects that they may be specified without reference to any particular individuals. Thus, e.g., it does not matter to the existential quantifier $\exists^e$ which elements the set characterized by its argument contains – but only whether it contains any elements at all. The fact that the exact identity of particular individuals does not matter means that, as far as the logical constants are concerned, all individuals are alike, they all play the same rôle – and are therefore interchangeable: trading individuals for each other would not have any effect on the result of a logical operation. This intuitive idea can be made precise by way of the notion of a *permutation*:[39] a permutation is a function that initially exchanges all individuals and then gets 'transferred' to objects of complex types:

(153) *Definition*

a. A permutation is a function π of type $\boldsymbol{ee}$ such that for any individuals x and x' the following holds:
   (i) if $x \neq x'$, then $\pi(x) \neq \pi(x')$;
   (ii) there is a z such that $x = \pi(z)$.

b. If π is a permutation and a is an extensional type, then π_a is a function of type aa so that the following holds:
   (i) if $a = \boldsymbol{e}$, then $\pi_a = \pi$;
   (ii) if $a = \boldsymbol{t}$, then $\pi_a = \{(0,0),(1,1)\}$;
   (iii) if $a = (bc)$ and f is a function of type a, then:
   $\pi_a(f) = \{(\pi_a(x), \pi_b(y))|\ f(x) = y\}$.

here.

[39] The idea to use permutations for the characterisation of logical constants was first proposed in the 1935 essay *On the Limitations of the Means of Expression of Deductive Theories* by Adolf Lindenbaum and Alfred Tarski, but appears to be so obvious that it has been repeatedly re-discovered in the meantime; its type-logical formulation goes back to a talk given by Alfred Tarski in 1966, the manuscript of which was published in 1986 under the title *What are Logical Notions?*.

Here, an *extensional* type is one that contains no $\boldsymbol{s}$; a generalization of permutations to all types, though possible, is somewhat awkward and unnecessary in the current context.

According to (153a), a permutation π must not only assign a unique individual to any individual – as any function of its type does; on top of that, π must never assign the same individual to two distinct arguments and, moreover, every individual must be assigned to some argument. If one conceives of functions as arrangements of arrows leading from argument to value, the two conditions express that no arrows may ever converge and that every individual gets hit by some arrow.[40] It should be noted that this definition does not exclude the limiting (but harmless) case of the identity mapping from x to x; it, too, is a permutation in the sense of (153a).

Basically, (153b) describes what happens to a function of an extensional type if all the individuals get permuted: permuted objects get mapped to permuted objects. As a case in point (the characteristic function of) the set $M = \{x, y, z\}$ of individuals gets mapped to (the characteristic function of) the set $M_\pi = \{\pi(x), \pi(y), \pi(z)\}$. In particular, M and M_π have the same cardinality. Since the existential quantifier only cares about the cardinality of its arguments, it will also map M and M_π to the same truth values $1 = \pi(1)$. Generalising this consideration, one sees that for any permutations π the following holds: $\pi(\exists^{\boldsymbol{e}}) = \exists^{\boldsymbol{e}}$. In this sense the existential quantifier does not 'notice' if any individuals have been exchanged:

(153) c. An object X of an extensional type a is *(permutation) invariant*, if $\pi_a(X) = X$, for all permutations π.

Apart form existential and universal quantifiers, the lexical determiner extensions considered in Chapter 3 come out as invariant, as well as all truth tables and the relation of identity between individuals (regarded as an object of type $\boldsymbol{e(et)}$). It can also be shown that the denotations of type-logical formulae (with or without identity) are always invariant if they contain no constants or free variables. Given that all logical constants can be defined by such formulae, their invariance thus follows.[41]

[40] In mathematical parlance, the first conditions says that permutations are *injective* or *one-one* and the second one says that they are *surjective* or *onto*. Functions that meet both conditions are also called *bijective*. A permutation, then, is a bijection on the set of individuals. As is readily seen, the inversion of a bijection π, that is, the set of all pairs $(\pi(x), x)$, is again a bijection.

[41] A similar connection between definability and permutation invariance holds for practically all languages of formal logic (with the exception of some that contain so-called *choice operators*). However, this generalization does not reverse: for any (countable) logical language there are invariant objects that cannot be defined in it.

5.8.2 Intensional type logic

Besides two-sorted type logic, the most frequent logical language used in the semantic literature for the purpose of indirect interpretation is the *intensional type logic* (or simply *Intensional Logic*) developed by Richard Montague.[42] Although this language does without any explicit reference to situations, it does preserve a large amount of the expressivity of two-sorted type logic.

The starting point of intensional type logic is a modification of the notion of a two-sorted type, based on the observation that, as far as extensions and intensions are concerned, Logical Space only acts as a functional domain: there are apparently no natural language expressions whose extensions are of a type of the form $\boldsymbol{as}$; moreover natural language does not seem to contain expressions of the type $\boldsymbol{s}$ itself. As a result, intensional type logic is restricted to types where $\boldsymbol{s}$ occurs to the left of another type. The primitive *intensional types* are, accordingly, only the types $\boldsymbol{e}$ and $\boldsymbol{t}$; and complex intensional types are either generated by taking pairs – thus ab is an intensional type if a and b are – or by prefixing $\boldsymbol{s}$ to a type – so that $\boldsymbol{s}a$ is an intensional type whenever a is. The objects of an intensional type a are then defined exactly as in two-sorted type logic: the objects of type $\boldsymbol{e}$ are the individuals; those of type $\boldsymbol{t}$ are the truth values; the objects of an intensional type ab are the functions that assign to all objects of (intensional) type a some object of (intensional) type b; and the objects of (intensional) type $\boldsymbol{s}a$ are the functions that assign to any possible situation an object of (intensional) type a.

As expected, intensional type logic contains infinitely many variables of any intensional type a, which – given an assignment – denote objects of type a. And as expected, intensional type logic also contains suitably many constants of any intensional type – which however receive a different interpretation than in two-sorted type logic. For, like their natural language counterparts, the constants of intensional type logic have both an extension and an intension. The type a of a constant – and a formula of intensional type logic in general – is their *extension type*; accordingly, their intension is of type $\boldsymbol{s}a$. This feature has immediate consequences for indirect interpretation: while we translated a noun like **Tisch** by a two-sorted formula of the form $\mathbf{T}(\boldsymbol{i})$ (where $\mathbf{T}$ is a constant of type $\boldsymbol{s(et)}$), the corresponding translation into intensional type logic is merely the constant $\mathbf{T}$ of type $\boldsymbol{et}$, whose extension is the set of all tables in the situation s at hand.

In intensional type logic, each formula has two semantic values: its extension and its intension. To denote them, we will use the same notation as in two-

[42] The language was first introduced in Montague's paper *Universal Grammar* (1970); later accounts usually refer to the extended version used in *The Proper Treatment of Quantification in Ordinary English* (1973).

sorted type logic, adding the situation at hand as a second super-script (on top of the assignment): the extension of a formula $\boldsymbol{\alpha}$ in a situation s and given the assignment g, is thus written as '$\|\boldsymbol{\alpha}\|^{g,s}$', and its intension (given the same assignment): '$\|\boldsymbol{\alpha}\|^{g}$'. The relation between extension and intension is defined in intensional type logic as it was defined (for German expressions) in direct interpretation: $\|\boldsymbol{\alpha}\|^{g,s} = \|\boldsymbol{\alpha}\|^{g}(s)$, for all $s \in LS$.

Like two-sorted type logic, intensional logic has the expressive means of functional application and $\boldsymbol{\lambda}$-abstraction, which work (on the level of extensions) in exactly the same way: $\|\boldsymbol{\alpha(\beta)}\|^{g,s} = \|\boldsymbol{\alpha}\|^{g,s}(\|\boldsymbol{\beta}\|^{g,s})$, and $\|\boldsymbol{(\lambda x.\alpha)}\|^{g,s}(u) = \|\boldsymbol{\alpha}\|^{g[x/u],s}$. Thus the sentence (154a), translated as (154b) into two-sorted type logic in Section 5.5, receives an analogous compositional translation (155) into intensional type logic:

(154) a. **Jeder Mann trifft eine Frau.** [= (91)]

b. $\boldsymbol{(\lambda Q.(\lambda P.\neg(\exists x)[Q(x) \wedge \neg P(x)]))(\mathrm{M}_i)}$ [= (92)]
$\boldsymbol{((\lambda x.(\lambda Q.(\lambda P.(\exists x)[Q(x) \wedge P(x)]))(\mathrm{F}_i)(\lambda y.\mathrm{T}_i(x,y))))}$

(155) $\boldsymbol{(\lambda Q.(\lambda P.\neg(\exists x)[Q(x) \wedge \neg P(x)]))(\underline{\mathrm{M}})}$
$\boldsymbol{((\lambda x.(\lambda Q.(\lambda P.(\exists x)[Q(x) \wedge P(x)]))(\underline{\mathrm{F}})(\lambda y.\underline{\mathrm{T}}(x,y))))}$

The intensional logic version (155) only differs from the original two-sorted translation (154b) at the three underlined places, where the explicit reference to a situation by the variable $\boldsymbol{i}$ has been replaced by the implict reference to a situation inherent to the intensional logic constant. And just like the two-sorted formula (154b), its intensional counterpart (155) can be equivalently reformulated by a sequence of $\boldsymbol{\lambda}$-reductions:

(156) $\boldsymbol{\neg(\exists x)[\mathrm{M}_i(x) \wedge \neg(\exists z)[\mathrm{F}_i(z) \wedge \neg\mathrm{T}_i(x,z)]]}$ = (98) [≡ (154b)]

(157) $\boldsymbol{\neg(\exists x)[\mathrm{M}(x) \wedge \neg(\exists z)[\mathrm{F}(z) \wedge \neg\mathrm{T}(x,z)]]}$ [≡ (155)]

The example suggests that intensional type logic leads to simpler (or at least shorter) formulae in indirect interpretation. We will, however, see that this only holds for extensional construction.

Functional application and abstraction of intensional type logic do not suffice to represent the complementation of an attitude verb. The extension of such a verb is of type $\boldsymbol{(st)(et)}$ and cannot be applied to the translation of the embedded clause, given that the latter is of type $\boldsymbol{t}$. What is obviously missing is a formula that stands for the intension of the embedded clause, rather than its extension. In two-sorted type logic we obtained this formula by $\boldsymbol{\lambda}$-abstraction from the situation variable $\boldsymbol{i}$; this option is not available in intensional type logic, simply because there is no such variable. Instead, at this point a so-called *cap operator* $^\wedge$ is invoked. Quite generally, this operator allows to construct an expression $\boldsymbol{(^\wedge\alpha)}$ of type $\boldsymbol{sa}$, starting with any expression $\boldsymbol{\alpha}$ of an (intensional) type a; and the extension of $\boldsymbol{(^\wedge\alpha)}$, then,

is $\boldsymbol{\alpha}$'s intension: $\|{\wedge}\boldsymbol{\alpha}\|^{g,s} = \|\boldsymbol{\alpha}\|^{g}$.[43]

Comparing the means of expression of intensional type logic introduced so far with those of two-sorted type logic, a simple connection emerges. Roughly, the two-sorted equivalent $\boldsymbol{\alpha^*}$ of an intension formula $\boldsymbol{\alpha}$, can be obtained by having $\boldsymbol{\alpha^*}$ express the situational dependence of $\boldsymbol{\alpha}$'s extension in terms of the variable '$\boldsymbol{i}$':

(158)

$\boldsymbol{\alpha}$	[formula of intensional type logic]	$\boldsymbol{\alpha^*}$	[two-sorted quivalent of $\boldsymbol{\alpha}$]
$\mathbf{c}$	[constant of an int. type a]	$\mathbf{c}^{+}\boldsymbol{(i)}$	[where $\mathbf{c}^{+}$ is a constant of type $\boldsymbol{s}a$]
$\boldsymbol{x}$	[variable of int. type a]	$\boldsymbol{x}$	
$\boldsymbol{\beta(\gamma)}$		$\boldsymbol{\beta^*(\gamma^*)}$	
$\boldsymbol{(\lambda x.\beta)}$		$\boldsymbol{(\lambda x.\beta^*)}$	
$\boldsymbol{({\wedge}\beta)}$		$\boldsymbol{(\lambda i.\beta^*)}$	

According to table (158), for any formula $\boldsymbol{\alpha}$ of intensional type logic (as considered so far) one can find a two-sorted formula $\boldsymbol{\alpha^*}$ whose denotation at any situation s is the extension of $\boldsymbol{\alpha}$, provided that $\boldsymbol{i}$ refers to s:[44]

(159) $\quad \|\boldsymbol{\alpha}\|^{g,s} = \|\boldsymbol{\alpha^*}\|^{g^*[i/s]}$

In (159), g is any assignment to the variables of intensional logic, whereas g^* is an assignment of all two-sorted variables that agrees with g on all intensional variables: $g \subseteq g^*$. Since, apart from $\boldsymbol{i}$, $\boldsymbol{\alpha}^*$ can only contain intensional variables, only their denotations and the value of $\boldsymbol{i}$ matter. The former are taken care of in (159) by g, the latter by the situation s to the left of the equality symbol.

Apart from the *cap operator*, there is a further construction in intensional type logic – its inverse, as it were – that is mainly needed to specify the semantic contributions of intensional arguments. Thus, the extension of the attitude verb **meinen** [≈ think] can be represented by a complex formula of two-sorted type logic that describes which rôle the intension of the comple-

[43] It should be noted that the extension of an expression of the form form $\boldsymbol{({\wedge}\alpha)}$ does not depend on the situation at hand; for the intension of an expression does not vary across Logical Space. There are, of course, possible situations s in which, say, the German word **Tiger** [≈ tiger] has a different intension in that the same form *of the variant of German spoken in s* is used to denote donkeys. But the extension of the *actual* German word **Tiger** contains those individuals that *are* tigers – and not those that *are called* **Tiger**; for the fact that it is called a **Tiger** does not make a member of the *equus asinus* group a *panthera tigris*. By the intension of **Tiger**, we mean the function that assigns to any possible situation the extension of the word **Tiger** in actual German.

[44] $\boldsymbol{\alpha^*}$ is also known as the two-sorted translation of $\boldsymbol{\alpha}$. The connection mentioned in (159), as well as the notation introduced in (158), originate with Daniel Gallin's book *Intensional and Higher-Order Modal Logic* (1975).

ment clause (= the argument $\boldsymbol{p}$) plays, viz. being 'evaluated' at the doxastic alternatives; this evaluation boils down to a functional application of the intension to a situation:

(160) $\boldsymbol{(\lambda p^{st}.(\lambda x^{e}.(\forall j)[\mathbf{DOX}(x)(i)(j) \rightarrow p_j]))}$ $|\mathbf{meint}|$, by (117a)

Since intensional logic contains no variables referring to situations, formulating analyses like (160) requires the means to express the application of an intension to a given situation. This is done by the so-called *cup operator* $^{\vee}$. Quite generally the operator allows to construct an expression $\boldsymbol{(^{\vee}\alpha)}$ of type a, starting from an expression $\boldsymbol{\alpha}$ of (intensional) type $\boldsymbol{s}a$; and the extension of $\boldsymbol{(^{\vee}\alpha)}$, then, is the value of the function denoted by $\boldsymbol{\alpha}$ for the situation at hand: $\|^{\vee}\boldsymbol{\alpha}\|^{g,s} = \|\boldsymbol{\alpha}\|^{g,s}(s)$. It should be noted that the cup operator can only precede formulae of types $\boldsymbol{s}a$, which themselves may have the form $\boldsymbol{(^{\wedge}\alpha)}$.[45] Obviously, the cup operator, too, easily transfers to two-sorted type logic:

(158)

$\boldsymbol{\alpha}$ [Formulas of int. type logic]	$\boldsymbol{\alpha^*}$ [two-sorted equivalent of $\boldsymbol{\alpha}$]
...	...
$\boldsymbol{(^{\vee}\beta)}$	$\boldsymbol{\beta^*(i)}$

Since cap and cup are notational variants of $\boldsymbol{\lambda}$-abstraction and functional application, they are also subject to a law of $\boldsymbol{\lambda}$-conversion, which is heavily used in indirect interpretation based on intensional type logic:

(161) *Down-Up-Cancellation*
If $\boldsymbol{\alpha}$ is a formula (of intensional type logic) of type a, the following holds:
$^{\vee}(^{\wedge}\boldsymbol{\alpha}) \equiv \boldsymbol{\alpha}$.

In view of the *-counterparts of intensional formulae of the form $^{\vee}(^{\wedge}\boldsymbol{\alpha})$, Down-Up-Cancellation proves to be a variant of *eigen*-conversion; for, as is readily seen by inspection of (158), $\boldsymbol{(^{\vee}(^{\wedge}\alpha))^* = (\lambda i.\alpha^*)(i)}$. The detour via two-sorted counterparts also explains why there is no mirror image law of Up-Down-Cancellation: $\boldsymbol{(^{\wedge}(^{\vee}\alpha))^* = (\lambda i.\alpha^*(i))}$, which is not necessarily equivalent to $\boldsymbol{\alpha}$ if $\boldsymbol{\alpha^*}$ contains the variable $\boldsymbol{i}$.[46]

Even with the cup operator, the analysis (160) of the attitude verb **meint** cannot be expressed in intensional type logic without further ado; for apart from the ubiquitous $\boldsymbol{i}$, it contains a further variable of $\boldsymbol{s}$, viz. '$\boldsymbol{j}$'. However, this is a bound variable, which can in principle be turned into a (bound) '$\boldsymbol{i}$' by bound renaming:[47]

[45] – but they need not: as a case in point, in the intensional version (165) of (160) it gets combined with a variable (of type $\boldsymbol{st}$).
[46] If $i \notin Fr(\boldsymbol{\alpha^*})$, then $\boldsymbol{(\lambda i.\alpha^*(i)) \equiv \alpha^*}$, by the principle (73e) of $\boldsymbol{\eta}$-conversion.
[47] The genearal strategy of eliminating bound variables of type $\boldsymbol{s}$ underlying (162) goes

(162) $(\boldsymbol{\lambda p^{st}.(\lambda x^{e}.(\forall j)[\mathbf{DOX}(x)(i)(j) \rightarrow p_j])})$ $\boldsymbol{\lambda}$-conversion
$\equiv$ $(\boldsymbol{\lambda p.(\lambda x.(\lambda q^{st}.(\forall j)[q_j \rightarrow p_j])(\mathbf{DOX}(x)(i))))}$ bound renaming
$\equiv$ $(\boldsymbol{\lambda p.(\lambda x.(\lambda q.(\forall i)[q_i \rightarrow p_i])(\mathbf{DOX}(x)(i))))}$

Thus reformulated, the analysis of **meint** [≈ thinks] can be expressed in intensional type logic, using (158), provided it contains the arrow '$\rightarrow$' and the universal quantifier. As before, the former can be defined by combining negation and conjunction, which – unlike the non-logical constants – are written and interpreted as in two-sorted type logic. The same holds for the universal quantifier of type ***(et)t***, which may be reduced to negation und existential quantifier in the familiar way and is also written '$\forall$'. On the other hand, the universal quantifier appearing in (162) is of type ***(st)t*** and called the *necessity* (or *box*) *operator*) in intensional logic;[48] it may be defined in terms of a corresponding existential quantifier know as the *possibility* or *diamond operator*). Unlike their two-sorted counterparts, the two operators receive their own notation in intensional logic:

(163) a. $\Diamond$ is a logical constant of type ***(st)t*** (in intensional type logic) such that the following holds:
$\|\Diamond\|^{g,s} = \lambda p.\vdash\downarrow p \neq \emptyset \dashv$ cf. (33b) in 5.3
[= that function of type ***(st)t*** that assigns to any sentence intension p the truth value 1 just in case p does not characterise the empty set] cf. (34b) in 5.3

b. If $\boldsymbol{\varphi}$ is an (intensional) formula of type $\boldsymbol{t}$, then the formula $\Diamond(^{\wedge}\boldsymbol{\varphi})$ is abbreviated as $\Diamond\boldsymbol{\varphi}$.

c. $\Box$ is (an abbreviation for) the formula: cf. (104) in 5.6
$\boldsymbol{\lambda p^{st}.\neg\Diamond\neg(^{\vee}p)}$
of type ***(st)t***.

d. If $\boldsymbol{\varphi}$ is a formula of of type $\boldsymbol{t}$, then
$\Box\boldsymbol{\varphi}$
is (an abbreviation for) the formula
$\Box(^{\wedge}\boldsymbol{\varphi})$.

A comparison of the pertinent definitions and notational conventions for predicate logic quantifiers reveals how the variable $\boldsymbol{i}$ is 'simulated' by cap and cup operators in intensional type logic. On the basis of (163) and (158) the following consequences for a two-sorted re-translation of the *modal operators* (= box & diamond) as quantifiers of type ***(st)t*** emerge:

(164) a. $(\Diamond\boldsymbol{\varphi})^{*} \equiv (\exists\boldsymbol{i})\boldsymbol{\varphi}^{*}$; b. $(\Box\boldsymbol{\varphi})^{*} \equiv (\forall\boldsymbol{i})\boldsymbol{\varphi}^{*}$.

back to my paper *Intensional Logic and Two-Sorted Type Theory* (1989).

[48] According to an idea that goes back to the German philosopher Gottfried Wilhelm Leibniz (1646–1716), necessity can be construed as truth in – or more accurately: of – all possible worlds. In modal logic, from where the notation introduced in (163) derives, this notion of necessity is dubbed *metaphysical* or *absolute necessity*.

Given (164), we can now transfer the (reformulated) analysis (162) of the attitude verb **meinen** into intensional type logic:[49]

(165) $(\boldsymbol{\lambda p^{st}.(\lambda x^{e}.(\lambda q^{st}.\Box[{}^{\vee}q \rightarrow {}^{\vee}p])(^{*}\mathbf{DOX}(x))))}$

A comparison of (160) and (165) brings out clearly that the formulae of intensional type logic are not always shorter and more transparent than their two-sorted counterparts. As soon as more than one situation gets into focus, the omission of the situation variable $\boldsymbol{i}$ may turn into a nuisance. And this is not just a matter of the readability of formulae but also has repercussions on their formal properties. For intensional logic violates a number of fundamental logical laws. One of them is the so-called *principle of specialization* – the (generalized) inference from a universally quantified formula $\boldsymbol{(\forall x)\varphi}$ to any instance $\boldsymbol{\varphi[x/\alpha]}$ in which the variable $\boldsymbol{x}$ has been replaced by a term $\boldsymbol{\alpha}$ of the same type – provided that $\boldsymbol{\alpha}$ contains no free variable that would thereby get bound. The (valid) formula $\boldsymbol{(\forall x^{e})(\exists y^{e})\Box(x = y)}$ of intensional logic is a counterexample to this principle: it does *not* guarantee the truth of $\boldsymbol{(\exists y^{e})\Box(\mathbf{c} = y)}$ if $\mathbf{c}$ is a constant (of type $\boldsymbol{e}$) whose extension is situation-dependent. Its two-sorted *-counterpart shows why this inference fails: the necessity operator binds the situation variable $\boldsymbol{i}$ implicit in the constant $\mathbf{c}$. In a similar way, the general validity of the principle of $\boldsymbol{\lambda}$-conversion can be refuted in intensional type logic: $\boldsymbol{(\lambda x^{e}.(\exists y^{e})\Box(x = y))(\mathbf{c})}$ is not equivalent to $\boldsymbol{(\exists y^{e})\Box(\mathbf{c} = y)}$. These logical principles (specialisation and $\boldsymbol{\lambda}$-conversion) only hold in restricted forms – namely, if the *-counterpart of the formula to be substituted does not contain a free $\boldsymbol{i}$ that contradicts the variable condition – producing an accidental binding by substitution.[50] But even when properly restricted will the principle of $\boldsymbol{\lambda}$-conversion wreak havoc: unlike their counterparts in two-sorted type logic, sequences of $\boldsymbol{\lambda}$-reductions (and renamings) starting with the same formula do not always lead to the same result.[51] Given this formal complication, it seems advisable to shun

[49] $^{*}\mathbf{DOX}$ is an intensional constant of type $\boldsymbol{s(et)}$ whose *-translation is equivalent to the following formula of two-sorted type logic:

$$\boldsymbol{\lambda j.\lambda x^{e}.\mathbf{DOX}(x)(i)(j)}.$$

[50] In this restricted form, the principles can also be formulated without reference to the two-sorted *-counterparts of intensional formulae. Following Gallin's proposal (cf. footnote 44), one may define the notion of a *modally closed* intensional formula, which covers such formulae in whose *-counterpart $\boldsymbol{i}$ does not occur freely. The modally closed formulae comprise (i) all variables, (ii) all formulae of the form $\boldsymbol{({}^{\wedge}\alpha)}$ as well as (iii) all functional applications $\boldsymbol{\alpha(\beta)}$ and $\boldsymbol{\lambda}$-abstractions $\boldsymbol{(\lambda x.\alpha)}$, where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are themselves modally closed. The principle of $\boldsymbol{\lambda}$-conversion can then be restricted (apart from the usual variable condition) so as to apply to $\boldsymbol{(\lambda x.\alpha)(\beta)}$ and $\boldsymbol{\alpha[x/\beta]}$ only if either $\boldsymbol{\beta}$ is modally closed or nowhere in $\boldsymbol{\alpha}$ does (free) $\boldsymbol{x}$ stand in the scope of a cap operator.

[51] This fact has been known since the publication of the paper *λ-Normal Forms in an Intensional Logic for English* (1980) by Joyce Friedman and David Warren, where an example along the following lines was presented: if $\mathbf{P}$ and $\mathbf{c}$ are constants of types $\boldsymbol{(se)e}$

intensional type logic and replace its formulae by their *-counterparts, even if the latter are sometimes longer.

5.8.3 Substitutional quantification

The interpretation of variables offered in Section 5.3 is not the only way of solving the problem of binding: one may also empoy *substitutions* instead of assignments. For this to work, one would have to assume, though, that any object u of any type a has a *standard name* $\mathbf{c}_u$, which is a (unique) constant of type a. Given this assumption, the denotation $\|\boldsymbol{\alpha}\|$ of a formula $\boldsymbol{\alpha}$ does not have to depend on an assignment. In particular, $\|\mathbf{c}_u\| = u$. The interpretation of the other constants proceeds as before; the variables, however, do not receive denotations of their own (or some that is arbitrarily chosen). Functional application, too, will be interpreted as before, but without reference to assignments. The $\boldsymbol{\lambda}$-operator, however, receives the following interpretive treatment:

(166) *Substitutional interpretation of* $\boldsymbol{\lambda}$*-abstraction*
If $\boldsymbol{x}$ is a variable of type a and $\boldsymbol{\alpha}$ is a formula of type b, then $\|\boldsymbol{(\lambda x.\alpha)}\|$ is that function of type (ab) such that for all u of type a the following holds:

$$\|\boldsymbol{(\lambda x.\alpha)}\|(u) = \|\boldsymbol{\alpha}[\boldsymbol{x}/\mathbf{c}_u]\|$$

As before, the notation '$\boldsymbol{\alpha}[\boldsymbol{x}/\mathbf{c}_u]$' stands for the replacement of any free occurrences of $\boldsymbol{x}$ by the constant c_u, the standard name of the object u. It should be noted that no accidental bindings can occur in (166), given that $\mathbf{c}_u$ contains no variables.

The substitutional interpretation of abstraction is independent of the two-sorted type logic considered here and can be applied to more or less all known forms of variable binding. In particular, one may also define a substitutional variant of intensional type logic. In practice, however, substitutional interpretation is chiefly found in connection with predicate logic.

(166) presupposes that the result of the substitution $\boldsymbol{\alpha}[\boldsymbol{x}/\mathbf{c}_u]$ has a denotation in the first place. If $\boldsymbol{x}$ is the only free variable in $\boldsymbol{\alpha}$, one can show that this is in fact so – even if $\boldsymbol{\alpha}$ (and hence $\boldsymbol{\alpha}[\boldsymbol{x}/\mathbf{c}_u]$) contains further $\boldsymbol{\lambda}$-operators. But if $\boldsymbol{\alpha}$ contains several free variables, $\boldsymbol{\alpha}[\boldsymbol{x}/\mathbf{c}_u]$ has no denotation (or, as the case may be, one that is totally random, due to the arbitrarily chosen denotations of free variables). In this sense, the technique of substitutional interpretation is intrinsically confined to closed formulae – which is no great loss: free variables and open formulae do not really have content;

and $\boldsymbol{e}$, respectively, then the formula $\boldsymbol{(\lambda x.\ \mathbf{P}((\lambda y.\ \hat{}y)(x)))(\mathbf{c})}$ can be (restrictedly) $\boldsymbol{\lambda}$-reduced to both $\boldsymbol{\mathbf{P}((\lambda y.\ \hat{}y)(\mathbf{c}))}$ and $\boldsymbol{(\lambda x.\ \mathbf{P}(\hat{}x))(\mathbf{c})}$, neither of which is further reducible or can be obtained from the other by bound renaming.

even accordng to assignement semantics, their denotations are provisional, assignment-dependent. However, in the realm of closed formulae, assignment semantics and substitutional interpretation agree.

According to (166), the denotation of a formula $(\boldsymbol{\lambda x.\alpha})$ – whether open or closed – is not determined from the denotations of its parts. Rather, the denotation depends on all substitution instances $\boldsymbol{\alpha}[\boldsymbol{x}/\mathbf{c}_u]$ none of which are part of the formula.[52] In contrast, as we have seen in Section 5.3, assignment semantics is compositional, at least on the level of meanings (= functions from assignments to denotations). Maybe it is its lack of compositionality that is chiefly responsible for the otherwise conceptually and technically simpler substitutional interpretation of variable binding. A further reason may be seen in the assumption of the somewhat artificial standard names that the substitutional interpretation of variable binding needs to rely on.[53]

[52] . . . apart from the neurotic case in which $\boldsymbol{\alpha}$ is closed (i.e., $Fr(\boldsymbol{\alpha}) = \emptyset$) and thus not even $\boldsymbol{x}$ is free in $\boldsymbol{\alpha}$.

[53] The substitutional approach to variable binding also leads to (surmountable) technical problems when combined with the (standard) model-theoretic approach to logic, where arbitrary (non-empty) sets may play the rôles of the domains of individuals and situations. – There is reason to suspect that the compositionality aspect was the crucial factor in the development of assignment semantics as an alternative of the older substitutional interpretation: it is known that Tarski (cf. Fn. 10) does not seem to have scruples regarding languages of arbitrary cardinality, which substitutional interpretation is committed to (and a source for qualms about it).

5.9 Exercises for Chapter 5

A1
1. What is the extension type of the meanings of coordinate conjunctions defined in (7) and (8)?
2. Give a reformulation of the compositional rule (4) so as to make it compatible with the analyses (7) and (8).

A2
1. Draw trees in the style of (20) for the formulae (19b) and (19c).
2. Undo the notational abbreviations in (26) and draw a tree for the result.

A3 Show that (25) $(\boldsymbol{\lambda Q^{et}.(\lambda P^{et}.(\exists x^{e})[Q(x) \wedge P(x)])})$ is a formula of type ***(et)((et)t)***.

A4 Represent $\|(\boldsymbol{\lambda x.\ S(i)(x)(y)})\|^{g_2}$ in the style of (43) by 'thinning' table (41).

A5 Reformulate the interpretation (47) of $(\boldsymbol{\lambda x^{a}.\ \alpha})$ as a combination of the meanings of $\boldsymbol{x^{a}}$ and $\boldsymbol{\alpha}$.
Hint: It suffices to construct the modified assignment $g[^{x}/_{u}]$ from g, u and the meaning of $\boldsymbol{x}$ – the denotation depending on the assignment – for arbitrary assingments g and objects u of type a.

A6 Determine the denotation $\|(\boldsymbol{\lambda y^{e}.\ (\lambda x^{e}.\ S(i)(x)(y))})\|^{g_1}$ of (36), starting from table (41).

A7 Show that the denotation of $(\boldsymbol{\lambda i.\ (\lambda y^{e}.(\lambda x^{e}.\ S(i)(x)(y)))})$ does not depend on the assignment.

A8
1. Show that $(\boldsymbol{\lambda x^{e}.\ (x = y)})$ is not equivalent to $(\boldsymbol{\lambda y^{e}.\ (x = y)[^{x}/_{y}]})$.
2. show that $(\boldsymbol{\lambda x.\ (\exists y)\ T_{i}(x)(y))(y)}$ is not equivalent to $(\boldsymbol{\exists y)T_{i}(y)(y)}$.

A9 Show that the bound renaming performed on (96)
$\neg(\exists x)[M_{i}(x) \wedge \neg(\boldsymbol{\lambda P.}(\exists x)[F_{i}(x) \wedge P(x)])(\boldsymbol{\lambda y.}T_{i}(x,y))]$
was necessary, by specifying an assignment given which the denotation of (96) differs from that of:
$\neg(\exists x)[M_{i}(x) \wedge \neg(\exists x)[F_{i}(x) \wedge (\boldsymbol{\lambda y.}T_{i}(x,y))(x)]]$

A10 Prove (105) by reference to the abbreviation conventions (104) and the interpretation of type logic given in Section 5.3.

A11 Show that the transitions in (109) are correct, under the assumption (given in the text) that $g(\boldsymbol{i}) = s$. You may presuppose that $\|\|N\|\|^{g} = [\![N]\!]^{s}$ and $\|\|VP\|\|^{g} = [\![VP]\!]^{s}$.

A12 Show that $|\textbf{Eike besitzt einen Porsche}| \equiv (\exists y)[P_{i}(y) \wedge B_{i}(e,y)]$.

A13 Show that the ditransitive verb **schulden** [≈ owe] is an opaque verb and provide a lexical analysis in the style of (141) based on the (approximate) synonymy with **geben müssen** [≈ must give] mentioned in the text.

Chapter 6

Modification

In this chapter we will take a look at constructions in which a constituent is optionally expanded (or *modified*) by an adjunct without thereby changing its category. We start with the analysis of (restrictive) relative clauses and (attributive) adjectives, which both modify nominal constituents, and later briefly turn to the modification of verbs and sentences by adverbs. Before doing the latter, we will discuss some foundational (type-) theoretic issues.

6.1 Relative Clauses

6.1.1 Ambiguity in Relative Clauses

There are several kinds or uses of relative clauses. Let us consider the following ambiguous (surface) sentences:

(1) **Die türkische Kursteilnehmerin, die in der zweiten Reihe sitzt, hat die Klausur bestanden.**
[≈ The female Turkish course participant [,] who is sitting in the second row[,] passed the test.]

To begin with, (1) may say about the (only) female Turkish course participant that she passed the test and moreover convey where this person is sitting. In this case we are dealing with an *appositive* reading of the underlined relative clause. [This is the reading that is normally marked by commas in the English translation.] In particular, there would have to be precisely one (1) female Turkish course participant according to this reading. On the other hand, (1) may also be used if more than one Turkish woman takes the course, as long as only one of them is sitting in the second row; and then a claim is made about the latter's test result. In this case we are dealing with a *restrictive* use of the underlined relative clause [which is marked by omitting the commas in the English translation]. The difference between the two readings of (1) can be made clear by the following paraphrases:

(1) a. **Die türkische Kursteilnehmerin, die übrigens in der zweiten Reihe sitzt, hat die Klausur bestanden.**
[≈ The female Turkish course participant, who is, by the way, sitting in the second row, passed the test.]

b. **Diejenige türkische Kursteilnehmerin, die in der zweiten Reihe sitzt, hat die Klausur bestanden.**
[≈ Precisely that female Turkish course participant who is sitting in the second row passed the test.]

Obviously (1a) is inappropriate if there are several (relevant) female Turkish course participants – even if only one of them is sitting in the second row.[1] Rather, (1a) must be understood as an assertion about the only Turkish woman in class. It appears that the adverb **übrigens** [≈ by the way] is only compatible with the appositive reading of the relative clause in which it occurs; it thus disambiguates. The same goes for **bekanntlich** [≈ as is well known], [unstressed] **ja** [≈ as is well known], and a bunch of further adverbs and particles. Reversely, modifying the definite article by -**jenig** [≈ precisely [that]] announces a relative clause that must be construed restrictively, as is confirmed by its incompatibility with adding **übrigens** [≈ by the way], **bekanntlich** [≈ as is well known], **ja** [≈ as is well known] etc.:

(1) c. ***Diejenige türkische Kursteilnehmerin, die übrigens in der zweiten Reihe sitzt, hat die Klausur bestanden.**
[≈ Precisely that female Turkish course participant who is, by the way, sitting in the second row passed the test.]

Not every relative clause is ambiguous in this way. As a case in point, (2) can only be understood appositively, i.e., in the sense of (2a); conversely, (3) lacks an appositive reading, which is again illustrated by its incompatibility (3*) with the above-mentioned disambiguating elements:

(2) **Selin, die in der zweiten Reihe sitzt, hat die Klausur bestanden.**
[≈ Selin, who is sitting in the second row, passed the test.]

a. **Selin, die ja in der zweiten Reihe sitzt, hat die Klausur bestanden.**
[≈ Selin, who is, by the way, sitting in the second row, passed the test.]

(3) **Keine türkische Kursteilnehmerin, die (*übrigens) in der zweiten Reihe sitzt, hat eine schlechte Klausur geschrieben.**
[≈ No female Turkish course participant who (*by the way) is sitting in the second row did bad on the test.]

In this chapter we will chiefly treat restrictive relative clauses and address the appositive ones only in passing. More specifically, we will neglect the

[1] As before we ignore the possibility of an anphoric use of the subject; we will return to anaphoric definite descriptions in Chapter 10 [which is yet to be written].

appositive *uses* of relative clauses; for the relative clause always has the same meaning, no matter if it is used restrictively or appositively.

6.1.2 Restrictive Relative Clauses

By way of approaching the analysis of restrictive relatives, let us look at:

(4) **Jede Kursteilnehmerin, die in der zweiten Reihe sitzt, hat die Klausur bestanden.**
[≈ Every female course participant who is sitting in the second row passed the test.]

(4) is apparently true of a given (classroom) situation s if the set A_s of female course participants in the second row in s is a subset of the set B_s of the successful attendants in s: $A_s \subseteq B_s$. The set A_s can then be constructed by intersecting the set K_s of female course participants (in s) with the set Z_s of persons sitting in the second row (in s): $A_s = K_s \cap Z_s$.Using this notation, the proposition expressed by (4) can then be represented as follows:[2]

(5) $\{s \in LS \mid K_s \cap Z_s \subseteq B_s\}$

Let us, for simplicity, assume that K_s, Z_s and B_s correspond to constants of type ***s(et)*** whose denotations, when applied to a given $s \in LS$, characterise these sets: $\|\mathbf{K}\| = \lambda s.\lambda x.\ \vdash x \in K_s\ \dashv$, etc. Then the type-logical representation of the extension of (4) corresponding to (5) looks like this:

(6) $(\forall \boldsymbol{x}^{e})[[\mathbf{K_i}(\boldsymbol{x}) \wedge \mathbf{Z_i}(\boldsymbol{x})] \rightarrow \mathbf{B_i}(\boldsymbol{x})]$

$\mathbf{K_i}$ is the translation of the noun **Kursteilnehmerin** [≈ female course participant], $\mathbf{B_i}$ translates the predicate **hat die Klausur bestanden** [≈ passed the test]. It is thus suggestive that the contribution the relative clause makes to the extension of (4) ought to be seen as consisting in the characteristic function of the set Z_s; we thus take it that its type-logical translation is equivalent to the formula $\mathbf{Z_i}$ – and thus to that of the predicate **sitzt in der zweiten Reihe** [≈ is sitting in the second row]. Precisely how the relative clause gets this semantic value will be made clear presently. Before that, however, we will ponder about how the extensions of the parts of (4) combine into the extension represented by (6). The answer to the underlined question is both simple and surprising. To begin with we observe that the subject is obviously a quantifier, so that the type-logical translation of the overall sentence S ought to be derivable by the translations of the subject Q and the predicate P, using the quantifier rule (88) from section 5.5: $|S| = |Q|(|P|)$. The translation of the predicate we have been taking to be: $\mathbf{B_i}$. The subject's translation is not hard to find either; for as we already remarked, the sentence expresses a subset relation between two sets A_s and

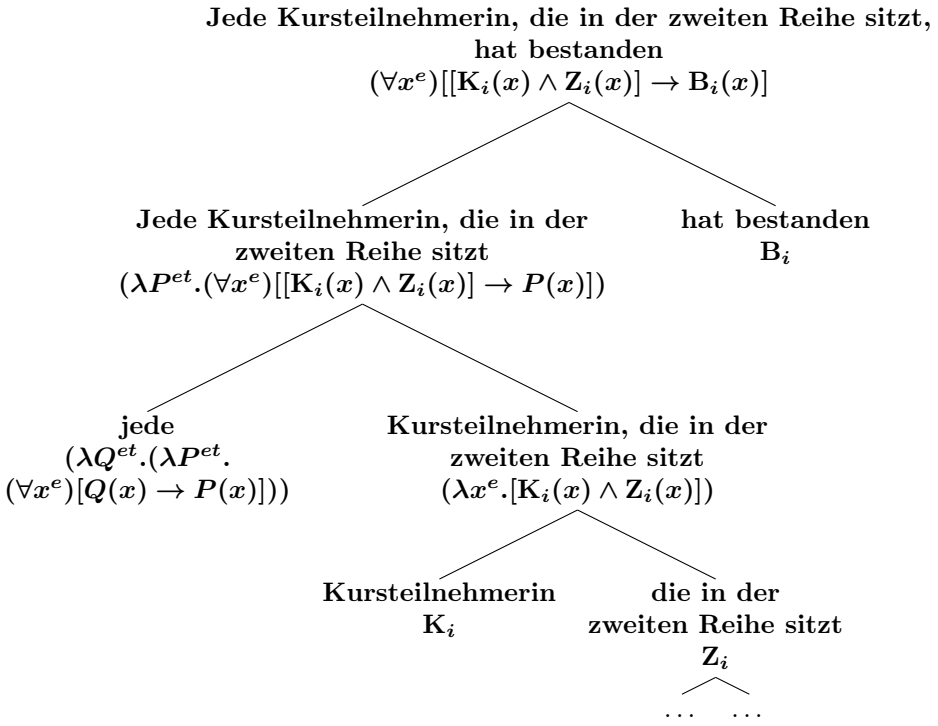
[2](5) thus is the set characterised by the intension of (4).

B_s, and this subset relation is just contributed by the determiner **jede** [≈ every] , whose extension accordingly must be applied to the intersection of the extension of **Kursteilnehmerin** [≈ female course participant] and that of the relative clause:

(7) $|\textbf{jede Kursteilnehmerin, die in der zweiten Reihe sitzt}|$
$\equiv$ $|\textbf{jede}|(\boldsymbol{\lambda x^e.}[|\textbf{Kursteilnehmerin}|(\boldsymbol{x}) \wedge$
$|\textbf{die in der zweiten Reihe sitzt}|(\boldsymbol{x})])$
$\equiv$ $(\boldsymbol{\lambda Q^{et}.}(\boldsymbol{\lambda P^{et}.}(\forall \boldsymbol{x^e.}[\boldsymbol{Q(x)} \rightarrow \boldsymbol{P(x)}]))\ (\boldsymbol{\lambda x.}[\mathbf{K}_i(\boldsymbol{x}) \wedge \mathbf{Z}_i(\boldsymbol{x})])$

It is now but a small step to a simple semantic analysis of attaching the relative clause *to the noun*:

(8)



The surprising thing about (8) is its bracketing: contrary to any (assumed) expectation, the relative clause does not go with the quantifying noun phrase **jede Kursteilnehmerin** [≈ every female course participant] but with the noun **Kursteilnehmerin** [≈ female course participant]. This analysis suggested by (7) also proves to be both handy and correct in other cases, as will be shown in an exercise (of course). And it cannot be escaped in the environment of a compositional semantic analysis – as will also be shown in an exercise, though a less simple one. The general pattern, then, is this:

(9) *Indirect Interpretation of the Attachment of Restrictive Relative Clauses*
If N is a (complex) noun consisting of a (possibly complex) noun N'

and a relative clause M, then:

$$|N| = (\boldsymbol{\lambda x^e.}[|N'|(\boldsymbol{x}) \wedge |M|(\boldsymbol{x})]).$$

Like the analysis in (8), the general rule (9) presupposes that the extension of the relative clause is of type $\boldsymbol{et}$. Now, how does the relative clause get an extension like that? To answer this question, we must attribute to it a rudimentary syntactic structure that takes the fact into account that a relative pronoun always relates to a gap in the remaining clause:

(10) a. **[die$_x$[__$_x$ in der zweiten Reihe sitzt]]**
[≈ [who$_x$[__$_x$ is sitting in the second row]]]

(10a) indicates the coarse structure of the relative clause from (4): the relative pronoun is combined with the *matrix* of the relative clause, a (subordinate) sentence with a gap – in our case in its subject position – that has the same subscript '$\boldsymbol{x}$'. This *coindexing* serves to uniquely identify the gap.[3]

(10) b. **[die$_x$[__$_x$ Fritz kennt]]**
c. **[die$_x$[Fritz __$_x$ kennt]]**

In (10a) and (10b) the (nominative) relative pronoun relates to the subject position of **kennt** [≈ knows]; in (10c) it is in the accusative and relates to its object position. Accordingly, the extension of (10b) should consist of the individuals who know Fritz, whereas the extension of (10c) should comprise those that Fritz knows:

(11) b. $|\textbf{die}_x[\_\__x\ \textbf{Fritz kennt}]|$
$= (\boldsymbol{\lambda x^e.}|\textbf{kennt}|(\boldsymbol{x}, |\textbf{Fritz}|))$
$= (\boldsymbol{\lambda x.}\mathbf{K}_i(\boldsymbol{x}, \mathbf{f}))$
c. $|\textbf{die}_x[\textbf{Fritz}\ \_\__x\ \textbf{kennt}]|$
$= (\boldsymbol{\lambda x^e.}|\textbf{kennt}|(|\textbf{Fritz}|, \boldsymbol{x}))$
$= (\boldsymbol{\lambda x.}\mathbf{K}_i(\mathbf{f}, \boldsymbol{x}))$

The translations given in (11b) and (11c) can be obtained in a simple and systematic way under the assumption that the gap is always translated by a variable (of type $\boldsymbol{e}$) the value of which is then abstracted from in the formation of the relative clause:

[3]The syntactic literature also has *traces* (*of movement*) instead of (*co-*) *indexed gaps*. – As a rule, a relative clause contains only one gap, which could also be identified without coindexing. But there are exceptions. Thus the accusative relative pronoun both stand for either of the two objects of **lehren** [≈ teach], with the other one being implicit: **die Gebiete, die niemand lehren will** [≈ the subjects no-one wants to teach] vs. **die Studenten, die niemand lehren will** [≈ the students no-one wants to teach]. Indexing removes ambiguity – provided that the two object positions are uniquely identified in the underlying syntactic structure. In (10) this is done by the 'canonical' order Subjekt Objekt Verb, which we have been assuming for simplicity; in general the positions are identified by detailed syntactic structuring.

(12) *Indirect Interpretation of Relative Clauses*
If M is a (subordinate) sentence and $\boldsymbol{x}$ is a variable of type $\boldsymbol{e}$, then:
$|\textbf{d-}_{\boldsymbol{x}}\ M| = (\boldsymbol{\lambda x^e.}|M|)$.

(13) *Indirect Interpretation of Indexed Gaps ('Traces')*
$|\_\__{\boldsymbol{x}}| = \boldsymbol{x}$ where $\boldsymbol{x}$ is a variable of type $\boldsymbol{e}$

Following this line, one now gets a translation of the relative clause in (10a):

(11) a. $|[\textbf{die}_{\boldsymbol{x}}[\_\__{\boldsymbol{x}}\ \textbf{in der zweiten Reihe sitzt}]]|$ by (12) and (13)
≡ $(\boldsymbol{\lambda x^e.}\ |\textbf{sitzt in der zweiten Reihe}|(\boldsymbol{x}))$ by assumption
≡ $(\boldsymbol{\lambda x.Z_i(x)})$ for type-logical reasons[4]
≡ $\boldsymbol{Z_i}$

Much can be said about the interpretive rules (12) and (13). First of all, it should be noted that (12) presupposes that the syntactic analysis already provides the coindexing of relative pronoun and gap. In particular, one may assume that the matrix M contains a (correctly indexed) gap in the first place. 'Empty' relativisation, as in **[der$_{\boldsymbol{x}}$ Fritz niemanden kennt]** [≈ **[who$_{\boldsymbol{x}}$ Fritz knows nobody]**] are thus excluded syntactically, although they would in fact make – limited – semantic sense (as is shown in an exercise). Apart from that, the coindexing must be made by way of a type-logical variable (of type $\boldsymbol{e}$). This is, of course, merely a matter of semantic convenience: one could have taken different indices and assigned variables to them instead.

It must furthermore be noted that (13) is a *lexical rule*: the indexed gap is understood as a simple expression.[5] In order for the gap to make its semantic contribution to the matrix, it must syntactically behave like a proper name and, in particular, be able to participate in subject and object predication. On the basis of the semantic equivalence of main and subordinate clauses familiar from Chapter 4, the matrices of the three relative clauses in (10) can now be translated into type logic:

(14) a. $|\_\__{\boldsymbol{x}}\ \textbf{in der zweiten Reihe sitzt}|$ reduction to verb second [main clause] word order
= $|\_\__{\boldsymbol{x}}\ \textbf{sitzt in der zweiten Reihe}|$ (Subject) Predication
= $|\textbf{sitzt in der zweiten Reihe}|(|\_\__{\boldsymbol{x}}|)$ by (13)
≡ $\boldsymbol{Z_i(x)}$
b. $|\_\__{\boldsymbol{x}}\ \textbf{Fritz kennt}|$ reduction to verb second [main clause] word order

[4]– by the so-called law of η-*Conversion*: cf. (73e) in section 5.4.

[5]There are reasons for assuming that gaps (or traces) are functional morphemes, which accordingly are not part of the lexicon. All that matters for our current purposes is that they are not complex expression whose semantic values must be reduced to those of their parts.

=	$\lvert \_\__x$ **kennt Fritz**$\rvert$	(Subject) Predication
=	**kennt Fritz**$\lvert(\lvert \_\__x \rvert)$	(Object) Predication
=	**kennt**$\lvert(\lvert$**Fritz**$\rvert)(\lvert \_\__x \rvert)$	by (13)
=	$\mathbf{K}_i(\mathbf{f})(x)$	notational convention
=	$\mathbf{K}_i(x, \mathbf{f})$	
c.	$\lvert$**Fritz** $\_\__x$ **kennt**$\rvert$	reduction to verb second [main clause] word order
=	$\lvert$**Fritz kennt**$\_\__x\rvert$	(Subject) Predication
=	**kennt** $\_\__x\lvert(\lvert$**Fritz**$\rvert)$	(Object) Predication
=	**kennt**$\lvert$ $(\lvert \_\__x \rvert)(\lvert$**Fritz**$\rvert)$	by (13)
=	$\mathbf{K}_i(x)(\mathbf{f})$	notational convention
=	$\mathbf{K}_i(\mathbf{f}, x)$	

Finally, (12) appears to be in conflict with the *Principle of Compositionality* in in the form assumed so far, and this for no fewer than three reasons. For one thing, the contribution the matrix makes to the extension of the relative clause consists neither in its extension nor in intension: the construction is thus neither extensional nor intensional. The reason for this is simply that according to (12), relative clause formation is a *binding operation*, where the assignment-dependence of the extension is reduced by binding a free variable. As we have seen (toward the end of section 5.3), this passage cannot be construed as an operation on the extension or the intension of the matrix; rather, the entire *meaning* of the matrix needs to be involved, i.e., the dependence of the extension on the variable assignment. If this meaning is itself taken as a function from assignments to extensions, variable binding – and relative clause formation according to (12) – can be conceived of as being *compositional on the meaning level*. We will not delve deeper into this but only point out that this is an essential property of relative clause formation (as well as certain other constructions to be considered in Chapter 7 [to be written]).[6]

But even if the Principle of Compositionality is weakened in this way, (12) does not seem to satisfy it. For there the (type-logically expressed) meaning of the relative clause is only traced back to the meaning of the matrix; the *relative pronoun* does not seem to contribute a meaning of its own. However, things are not that simple. In fact, (12) can be reconciled with the Principle of Compositionality (on the meaning level) – in rather different ways:

- (12) can be construed as saying that the relative pronoun – unlike the indexed gap – is no morpheme on its own but only indicates the grammatical construction in use (viz., relative clause formation); the relative pronoun would then be *syncategorematic*. On this view, its

[6]This only holds for the (standard) analysis of relative clauses pursued here. Compositionality on the level of extensions can be upheld in the framework of *variable-free interpretation* (as indicated at the end of the previous chapter).

status would be similar to that of the brackets or the lambda in type-logical notation, neither of which have meanig on their own, although they do contribute to the meanings of the formulae in which they occur. Furthermore, relative clause formation would then be a *unary* construction, i.e., not a combination of *several* expressions but one that merely one expression undergoes, viz., the matrix. And (12) would indeed interpret this construction in a compositional way; for the meaning of the entire expression (the relative clause) is systematically obtained from the meaning of its only part (the matrix), viz., by abstraction from the variable assignment.[7]

- In view of this, one may as well identify the semantic contribution of the relative pronoun with the meaning of the corresponding variable – and thus also with the meaning of the coindexed gap. The translation given in (12) could then be reformulate accordingly:

  $|\mathbf{d}\text{-}_{\boldsymbol{x}}\ M| = (\boldsymbol{\lambda}|\mathbf{d}\text{-}_{\boldsymbol{x}}|.|M|)$,

  where a further (lexical) translation rule has it that $|\mathbf{d}\text{-}_{\boldsymbol{x}}| = \boldsymbol{x}^e$. According to this analysis, the semantic contribution of relative clause formation thus consists precisely in the abstraction indicated by the type-logical $\boldsymbol{\lambda}$.

- One may also consider $\boldsymbol{\lambda}$-abstraction itself – or more precisely: the corresponding semantic operation – the meaning of the relative pronoun. For this one would, however, step out of the type-logical framework defined so far. For abstraction combines the meaning of the variables $\boldsymbol{x}$ with the meaning of the matrix M into the meaning of the $\boldsymbol{\lambda}$-formula; it is thus no meaning itself, i.e., no assignment-dependent extension.

We do not have to decide on any of the three compositional accounts here. It should merely be noted that, from a semantic point of view, relative clause formation differs from most of the constructions considered in the previous chapters in that it is not interpreted by type-logical functional application.[8] The same holds for the attachment of the restrictive relative clause to the noun, which in (9) is interpreted by *intersection*, set-theoretically speaking. We will, however, come across an alternative to (9) in Section 6.3.

[7] Since different variables correspond to different semantic operations, one would, strictly speaking, be dealing with a whol 'family' of constructions – one construction per variable; such a conception of relative clauses can be found in Richard Montague's writings.

[8] According to the third of the above possibilities relative clause formation does correspond to some kind of functional application, but not to that of typelogic; for no extension (of some type ab) gets applied to an extension (of some type a) or an intension (of some type $\boldsymbol{s}a$).

6.1.3 Appositive Relative Clauses

The attachment of *appositive* relative clauses, on the other hand, can be interpreted by functional application, although not in the by now familiar way. First of all, one should remember that this construction also occurs in connection with proper names. It is thus not (or not always) compatible with the constituent structure of the restrictively used relative clause. Thus in (2) the relative clause cannot modify a sortal noun; for the subject does not contain any such noun (outside the relative clause). We will assume the following bracketing instead:

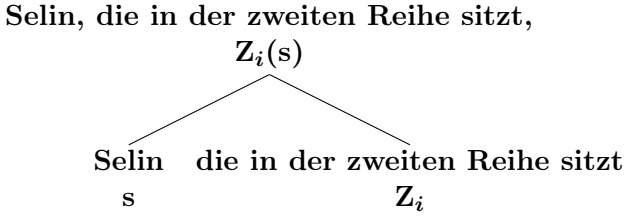
(2) **[[Selin, [die in der zweiten Reihe sitzt,]] hat die Klausur bestanden]**
[≈ [[Selin, [who is sitting in the second row,]] passed the test]]

Under the assumption that the meaning of the relative clause is independent of its (restrictive vs. appositive) attachment, the type-logical translations of the three bracketed constituents in (2) may be presumed to be known:

(15) a. $|\textbf{Selin}| = \mathbf{s}$, constant of type $\boldsymbol{e}$
b. $|\textbf{die in der zweiten Reihe sitzt}| \equiv \mathbf{Z}_{\boldsymbol{i}}$, where $\mathbf{Z}$ is a constant of type $\boldsymbol{s(et)}$
c. $|\textbf{hat die Klausur bestanden}| \equiv \mathbf{B}_{\boldsymbol{i}}$, where $\mathbf{B}$ is a constant of type $\boldsymbol{s(et)}$

If one now wants to combine these three type-logical formulae in parallel to the structure given in (2), functional application suggests itself for the interpretation of the subject (= proper name + relative clause):

(16)



It is not hard to see that the result of this combination is of type $\boldsymbol{t}$ and thus corresponds to a truth value (extensionally) or a proposition (intensionally). This is problematic in that there is no natural way for combining these semantic values with those of the predicate in (15c), which is of type $\boldsymbol{et}$ after all. It rather seems as if this predicate (or more precisely: its semantic values) must be combined with the proper name **Selin** by functional application – just like the relative clause:

(17) a. $\mathbf{Z}_{\boldsymbol{i}}(\mathbf{s})$
b. $\mathbf{B}_{\boldsymbol{i}}(\mathbf{s})$

On the *extensional layer* the resulting truth value obviously needs to be 1 in both cases, should the entire sentence (2) come out true. The corresponding intensions – this is not hard to see either – then have a different status:

(18) a. $\mathbf{(\lambda i.Z_i(s))}$
b. $\mathbf{(\lambda i.B_i(s))}$

(2) seems to *express* these two propositions at once, but *present* them in different ways. Whereas (18a) makes up the true informational value of (2), (18b) gives some information aside, which may either be presupposed to be known or marked to be less relevant. The compositional derivation of the meaning of (16) must account for this asymmetry and make a split between major and minor parts; a theoretical frame for this will be developed in Chapter 9 [which still needs to be written].

Appositive relative clauses not only occur with proper names. Definite descriptions, too, can be modified by them, as the intial example of this chapter has already illustrated:

(19) **Die türkische Kursteilnehmerin, die in der zweiten Reihe sitzt, hat die Klausur bestanden.** = [(1)]
a. **Die türkische Kursteilnehmerin, die übrigens in der zweiten Reihe sitzt, hat die Klausur bestanden.**
[≈ The female Turkish course participant, who is, by the way, sitting in the second row, passed the test.]
b. **Diejenige türkische Kursteilnehmerin, die in der zweiten Reihe sitzt, hat die Klausur bestanden.**
[≈ Precisely that female Turkish course participant who is sitting in the second row passed the test.]

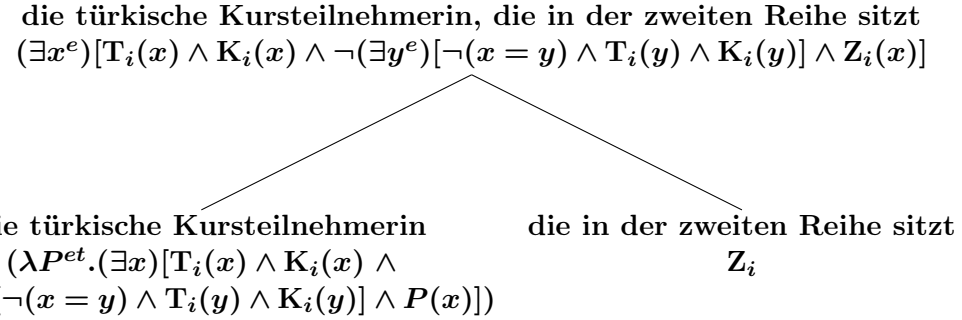
In the light of the above observations it is now tempting to explain the reading (19a) in analogy to the interpretation of (2): the definite description **die türkische Kursteilnehmerin** [≈ the female Turkish course participant] can be (syntactically) combined directly with the relative clause and (semantically) be interpreted as a claim that complements the proper content (or the proposition expressed). The difference between the restrictive and the appositive reading then comes out as a bracketing ambiguity:

(20) a. **[[[Die [türkische Kursteilnehmerin]] [die in der zweiten Reihe sitzt]] hat die Klausur bestanden]**
[≈ [[[The [female Turkish course participant]] [who is sitting in the second row]] passed the test]]
b. **[[Die [[türkische Kursteilnehmerin] [die in der zweiten Reihe sitzt]]] hat die Klausur bestanden]**
[≈ [[The [[female Turkish course participant] [who is sitting in the

second row]]] passed the test]]

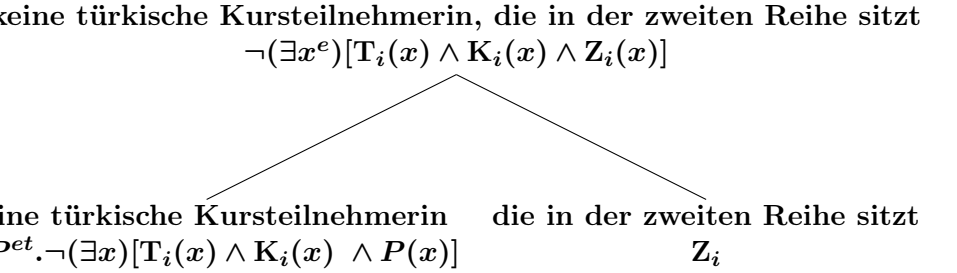
(There is a special exercise dedicated to those readers who find these brackets too confusing!) As in the case of (16), the combination of the definite description with the relative clause can be interpreted by functional application, but unlike in that case, the extension of the definite description must be applied to that of the relative clause: following Russell's interpretation of the definite article, definite descriptions are quantifying noun phrases after all:

(21)



Intuitively speaking, the relative clause in (21) has the same function as that in (16), viz., providing additional information about the person denoted by its sister constituent. It is thus not surprising that appositive relative clauses cannot modify quantifying noun phrases like **keine Kursteilnehmerin** [≈ no female course participant] (as we had already seen by way of (3)): unlike definite descriptions, 'true' quantifiers do not denote individuals. This insight is not reflect in (21) though, since the meaning combination used for interpreting relative clause attachment would just as well work with 'true' quantifiers; as a case in point, the subject of (3) would receive the following appositive interpretation:

(22)



However, on no reading does (3) say – and be it only as an aside – that the second row did not contain any female Turkish course participants. The appositive construction (22), though parallel to (21), must therefore be blocked somehow – perhaps by a pertinent syntactic restriction. A more natural explanation emerges if one follows the above-mentioned intuition that definite descriptions denote the individuals they describe like proper names denote

their bearers. For if the semantic value of a definite description were an individual, definite descriptions would not be quantifiers, and there would be analogy between (1) and (3). The bracketing (20a) of (1) could then be interpreted in analogy to (2). We will return to this topic in Chapter 9 [to be written].

6.2 Adjectives

In (21) and (22) we tacitly assumed the complex noun **türkische Kursteilnehmerin** [≈ female Turkish course participant] to translate as follows:

(23) $\boldsymbol{(\lambda x^{e}.[\mathbf{T}_{i}(x) \wedge \mathbf{K}_{i}(x)])}$

Of course, both (non-logical) constants appearing therein are of type ***s(et)***:[9]

(24) a. $\|\mathbf{T}\| = \lambda s.\lambda x. \vdash x$ is of Turkish nationality in $s \dashv$
b. $\|\mathbf{K}\| = \lambda s.\lambda x. \vdash x$ is a female course participant in $s \dashv$

Obviously, to obtain (23) as the desired result of the indirect interpretation of **türkische Kursteilnehmerin** [≈ female Turkish course participant], one first needs to equate the intensions of the adjective **türkisch** [≈ Turkish] and the noun **Kursteilnehmerin** [≈ female course participant] with the respective denotations of the constants interpreted in (24a) and (24b):

(25) a. $|\mathbf{türkisch}| = \mathbf{T}_{i}$
b. $|\mathbf{Kursteilnehmerin}| = \mathbf{K}_{i}$

. . . and combine the results by intersection – just as in the case of the attachment of the restrictive relative clauses to their head nouns:[10]

(26) *Indirect Interpretation of Attaching Intersective Adjectives*

[9] (24b) raises the question which course is referred to to the right of the equality sign. A straightforward answer is that this depends on the situation s at hand, i.e., that it is the *course that is pertinent in s*. An alternative answer relates the question to context dependence, the topic of the next chapter but one [which is not yet written]. We leave the question open for the time being. Moreover we will neglect the fact that the adjective **türkisch** [≈ Turkish] is obviously polysemous in a systematic way and apart from nationality, may also relate to the place of residence or the origin.

[10] The formula given in (26) only describes an intersection if neither the translation of the noun nor that of the adjective contains $\boldsymbol{x}$ as a free variable. Otherwise undesired binding occurs, which may be avoided by the somewhat more cumbersome formula:

(*) $|N| = \boldsymbol{(\lambda Q^{et}.(\lambda P^{et}.(\lambda x^{e}.[Q(x) \wedge P(x)])))}\ (|N'|)\ (|A|)$

Clearly, for the case that $\boldsymbol{x} \notin Fr(|N'|) \cup Fr(|A|)$, (*) is equivalent to the translation given in (26).

If N is a (complex) noun consisting of an intersective adjective A and a (possibly complex) noun N', then:

$$|N| = (\boldsymbol{\lambda x^e}.[|N'|(\boldsymbol{x}) \wedge |A|(\boldsymbol{x})])$$

Apart from the position and the categories of the constituents involved, (26) resembles the semantics (9)of relative clause attachment; in both cases the intersection of the extensions (conceived as sets) is formed. This, of course, presupposes that the extension of the adjective can be construed as a set in the first place. In the case of **türkisch** [≈ Turkish] this is a straightforward assumption – by adding the adjective the extension of the noun obviously gets restricted to individuals with Turkish nationality. But the combination of an adjective and a (sortal) noun cannot always be described along the lines of (26):

(27) **Ein junger Anwalt verteidigt einen angeblichen Hochstapler.**
[≈ A young lawyer is defending an alleged conman.]

(27) is true of situations in which there are no conmen at all; all it takes is someone who is alleged to be one. Moreover, it is irrelevant for the truth of (27) whether there are any necrophiliacs in the situation at hand. And it is easy to imagine situations to which (27) applies and in which there are neither conmen nor necrophiliacs and no one is alleged of desecreting corpses. In particular, the following sentence does not apply to such situations:

(28) **Ein junger Anwalt verteidigt einen angeblichen Leichenschänder.**
[≈ A young lawyer is defending an alleged necrophiliac.]

Since (27) and (28) may differ in their truth values in the same situation, **angeblich** [≈ alleged] cannot be an intersective adjective. For the extensions of **Hochstapler** [≈ conman] and **Leichenschänder** [≈ necrophiliac] coincide in such a situation – and thus so do those of [A **Hochstapler**] and [A **Leichenschänder**] if A is an adjective interpreted according to (26), as is easily shown in an exercise.

A similar argument shows that the other adjective in (27) and (28) cannot be interpreted by intersection either.[11] For, as will become clear by solving one of the exercises (if not earlier), if **jung** [≈ young] were an intersective adjective, the young politicians would have to form a subset of the young lawyers in a situation in which all politicians happen to be lawyer. But even under such circumstances the truth of (27) does not necessarily imply that (29) is also true:[12]

[11] Thanks to Dina Voloshina for spotting a flaw in the example used in an earlier versions of these class notes.

[12] It ought to be mentioned that, according to a view that is common among semanticists,

(29) **Ein junger Politiker verteidigt einen angeblichen Hochstapler.**
[≈ A young politician is defending an alleged conman.]

(27) may apply to a situation if said interviewer is *young for a politician* without being at the same time *young for a lawyer.* If, e.g., the average age for politicians in a certain country at a certain time were 63, a 55-year old party leader would still be a comparatively young politician, but she would no longer be a young lawyer if the average age of that occupational group was around 50 years. In other words: being a young politician is not the same as being a politician who is young – in an absolute sense; rather, a young politician is someone who is both a politician and young for a politician. In the case of adjectives like **jung** [≈ young], then, the set with which the extension of the modified noun gets intersected thus seems to depend on which noun it is. Under the assumption that someone is young relatively to a comparison group if his or her age is clearly below the average age of that group, the extension of a complex noun of the form **jung-** N [≈ **young** N] can be determined by intersecting the extension of the sortal N with the set of persons whose age lies clearly below the average age of the elements in the extension of N. As in the case of an intersective adjective, then, the adjective's contribution to the extension determined is a set of individuals which gets intersected with the extension of the head modified; but in the case of adjectives like **jung** [≈ young] – as opposed to those like **türkisch** [≈ Turkish] – this set also depends on the extension of the modified noun. Since the set with which the extension of the noun must be intersected depends on this extension itself, the extension of an adjective like **jung** [≈ young] cannot itself be an object of type ***et*** (a set of individuals). Rather, it must be a set depending on a noun extension – i.e., a set-dependent set, and thus an object of type ***(et)(et)*** – which in the course of modifying a noun is both applied to the latter's extension and intersected with it. We will call such adjectives *extensionally subsective.* Their contribution to the modification of nominal extensions can be described as follows:

(30) *Indirect Interpretation of Attaching Extensionally Subsectives Adjectives*
If N is a (complex) noun consisting of an extensionally subsective adjective A and a (possibly complex) noun N', then:
$$|N| = (\boldsymbol{\lambda x^e.}[|N'|(\boldsymbol{x}) \wedge |A|(|N'|)(\boldsymbol{x})]).$$

the difference between (27) and (28) is of a pragmatic nature. In this view, the adjective **jung** [≈ young] is intersective but highly *context-dependent*: its extension consists of those individuals whose age is below some given standard, where the *use* of one or the other modified sortal – **Anwalt** [≈ lawyer] vs. **Politiker** [≈ politician] – makes such a standard salient. We will briefly return to this analysis in the next chapter but one [which still needs to be written] but ignore it for the time being – mainly for didactic reasons.

An indirect interpretation of the adjective **jung** [≈ young] under scrutiny then looks like this:

(31) a. $|\mathbf{jung}| = \mathbf{J}_{\boldsymbol{i}}$, where $\mathbf{J}$ is a constant of type $\boldsymbol{s((et)(et))}$.
b. $\|\mathbf{J}\| = \lambda s.\lambda M.\lambda x.\vdash \overline{\alpha_s^M} \gg \alpha_s(x) \dashv$, where $\alpha_s(x)$ is the age of the individual x in the situation s and $\overline{\alpha_s^M}$ is the average value of the age function α_s over the elements of the set M.[13]

Adjectives like **jung** [≈ young] are called *extensionally* subsective because the intersection they contribute to determining the *extension* only depends on the extension of the noun modified. For if two nouns have the same extension, then the average age within this extension is the same and thus so are the extensions of their modification by **jung** [≈ young] as determined by (30) and (31). Things stand differently in the case of an *intensionally* subsective adjective like **begabt** [≈ gifted], where an intersection with the extension of the modified noun N is formed too though it does not depend on the extension of N but on its intension; for even if all sculptors happened to be engravers and vice-versa, a gifted sculptor still would not have to be a gifted engraver. Hence (30) cannot be applied to the constellation **begabt**-N [≈ gifted N]; instead we need:

(32) *Indirect Interpretation of Attaching Intensionally Subsective Adjectives*
If N is a (complex) noun consisting of an intensionally subsective adjective A and a (possibly complex) noun N', then:
$$|N| = \boldsymbol{(\lambda x^{e}.[|N'|(x) \wedge |A|(\lambda i.|N'|)(x)])}.$$

This time the type of the adjective is $\boldsymbol{(s(et))(et)}$.[14] However, in this case it is much harder to find a general rule determining the extension of the adjective. We content ourselves with a partial characterisation:

(33) a. $|\mathbf{begabt}| = \mathbf{B}_{\boldsymbol{i}}$, where $\mathbf{B}$ is a constant of type $\boldsymbol{s(s(et))(et)}$.
b. For any $s \in LS$ and any P of type $\boldsymbol{s(et)}$ the following hold:
if there is an activity such that for all $s' \in LS$ and all individuals x satisfy:
$P(s')(x) = 1$ iff x regularly performs this activity,
then for all $s' \in LS$ and all individuals x: $\|\mathbf{B}\|(s)(P)(x) = 1$ iff in s, x proves to be gifted in performing said activity.

[13] In other words: $\overline{\alpha_s^M} = \frac{1}{\overline{\overline{M}}} \cdot \sum_{y \in M} \alpha_s(y)$, where $\overline{\overline{M}}$ is again the number of elements of the set characterised by M, which we take to be finite for simplicity. '$n \gg m$' stands for the admittedly vague claim that the number n clearly exceeds m. Of course, (31) assumes a homogeneous numerical age measure – e.g., in (fractions of) seconds, years, etc.

[14] – not to be confused with the intension type $\boldsymbol{s((et)(et))}$ of extensional adjectives like **jung** [≈ young]!

(33b) appears complicated, but becomes accessible when applied to a specific example. If, e.g., we choose P to be the intension of the noun **Gitarrist** [≈ guitarist], we may first observe that the underlined condition in (33b) is indeed met: $P(s')(x)$ is 1 precisely if x is a guitarist in s', i.e., regularly plays the guitar; so the activity in question is guitar-playing. According to (33b), applying the extension of **begabt** [≈ gifted] to the intension of **Gitarrist** [≈ guitarist], then, (in a given situation s) results in (the characteristic function of) the set of those individuals x who display a certain talent in performing this activity – in other words: the set of gifted guitarists. So at least in this case (33) seems to work correctly.

(33b) says that the denotation of the constant **B** is a function of a particular type and what the values of this functions for particular arguments are. This leaves open what happens if the extension of **B** is applied to a P that does not meet the underlined condition – the intension of **Tisch** [≈ table], say. Moreover, it is by no means always clear that the characterisation of the functional values is always unequivocal even where the underlined condition is satisfied. Thus, e.g., a (Catholic) priest regularly performs certain activities that are reserved to (Catholic) priests alone. But then even if someone is a priest just in case he ⌊*sic!*⌋ regularly holds services, someone who does this with an exceptional talent still does not have to be a gifted priest. The example might not be fully convincing: after all, there might be priests that do not regularly perform any of the activities characteristic of this professional group. It still points to a hidden assumption in (33b) that may not be entirely unproblematic – viz., that the activity evaluated is uniquely determined by the intension of the noun. It is, however, unclear whether **begabt** [≈ gifted] could also be used if this requirement is not met.

Incomplete characterisations of semantic values, as in (33b), are called *meaning postulates*. They are invoked whenever spelling out all details of a lexical meaning is irrelevant, cumbersome, or difficult, while at the same time some of its properties need to be fixed. The partial characterisation of the intension of the noun **Junggeselle** [≈ bachelor] is a classical case in point:[15]

(34) a. $|\mathbf{Junggeselle}| = \mathbf{G}_{\boldsymbol{i}}$, where **G** is a constant of type ***s(et)***.
b. For any $s \in LS$ and any individual x the following holds:
if $\|\mathbf{G}\|(s)(x) = 1$, then x is s an unmarried mal adult person in

[15] *Meaning Postulates* was the title of a paper by Rudolf Carnap published in 1952, from where the term originates. – In the context of indirect interpretation meaning postulates are usually written as type-logical formulae. (34b) would then be replaced by:

(*) $(\forall \boldsymbol{i})(\forall \boldsymbol{x}^{e})[\mathbf{G}_{\boldsymbol{i}}(\boldsymbol{x}) \rightarrow [\neg|\mathbf{verheiratet}|(\boldsymbol{x}) \wedge |\mathbf{Mann}|(\boldsymbol{x})]]$

A corresponding type-logical representation of (33b) would lead to far astray.

s.

According to (34) the extension of **Junggeselle** [≈ bachelor] only contains unmarried men, which leaves open which ones precisely: all of them, including widowers, the pope. ...? Despite this sketchiness a postulate like (34b) may prove handy – e.g., when it comes to explaining the weirdness of sentences like (35):

(35) **Fritz kennt einen Junggesellen, der verheiratet ist.**
[≈ Fritz knows a bachelor who is married.]

In a similar way, one may exploit (33b) to explain why (36) sounds outlandish:

(36) **Fritz kennt einen begabten Pianisten, der des Klavierspielens nicht mächtig ist.**
[≈ Fritz knows a gifted pianist who is not capable of playing the piano.]

The meaning postulates (33b) and (34b) serve to capture at least some semantic aspects of the two lexemes under scrutiny. In both cases a full-fledged specification of the intension does not seem to be easy – albeit for different reasons. In the case of **Junggeselle** [≈ bachelor] it is hard to identify clearly sufficient conditions under which someone belongs in the extension; in particular, it is unclear which criteria make the literal meaning of and which are but pragmatic frills. Such problems of conceptual determination are typical for the lexical domain and show that the assumed reconstruction of meanings as functions in Logical Space is an extreme idealisation made by Logical Semantics. The problems mentioned above in connection with the specification of the meaning of **begabt** [≈ gifted] are of a different nature though, in that they do not only concern the incompleteness of the partial description (33b) of the intension but its correctness.[16] Such problems in turn are not quite atypical of the use of meaning postulates in logical semantics and are often indicative of a mistake in the analysis of a construction or in the assignment of an extension type. In Section 6.5 [to be written] we will come across an alternative analysis of adjectives like **begabt** [≈ gifted] that can do without a postulate like (33b).

At this point, however, we will turn to the above adjective **angeblich** [≈ alleged], the semantic analysis of which is still due. As one can easily see, it does not fall into any of the three above adjectival categories; for according to the construction interpretations (26), (30) and (32) the extension of a complex noun of the form $A\ N$ (construed as a set) is always a subset of the extension of the modified noun N – no matter whether A is intersective or

[16] ... which does not exclude that there are problems of conceptual determination on top of this – like the question of what it means that someone pursues an activity (sufficiently) regularly.

(intensionally or intensionally) subsective. But an alleged conman does not have to be a conman. Rather, an alleged conman is someone who is alleged to be a conman. Thus the following two sentences share the same intension:

(37) a. *NN* **ist ein angeblicher Hochstapler.**
[≈ *NN* is an alleged conman.]
b. **Es ist angeblich so, dass** *NN* **ein Hochstapler ist.**
[≈ It is allegedly the case that *NN* is a conman.]

In order to guarantee this intensional identity for arbitrary names *NN*, we will first analyse (37b). The laborious formulation in terms of clausal embedding ought to help interpreting the construction by Hintikka Semantics. For like the impersonal construction with **scheinen** [≈ seem] considered in the previous chapter, the clause-embedding use of **angeblich** [≈ alleged] can be interpreted by way of an alternative relation[17] of type $\boldsymbol{s(st)}$, where we keep notation and terminology as parallel as possible:

(38) a. $|\textbf{schein-}| = (\boldsymbol{\lambda p^{st}.(\forall j)[\mathbf{EVI}^{*}_{i}(j) \rightarrow p_j]})$ [= 5.7, (130)]
b. $|\textbf{angeblich}| = (\boldsymbol{\lambda p^{st}.(\forall j)[\mathbf{ASS}^{*}_{i}(j) \rightarrow p_j]})$

(39) a. $\|\mathbf{EVI}^{*}\| = \lambda s_0.\lambda s_1.\, \vdash s_1$ is an evidenial alternative in s_0 $\dashv$ [= 5.7, (127)]
b. $\|\mathbf{ASS}^{*}\| = \lambda s_0.\lambda s_1.\, \vdash s_1$ is an assertoric alternative in s_0 $\dashv$

The constant $\mathbf{ASS}^{*}$ used in the analysis (38b) of **angeblich** [≈ alleged] is meant to be based on a subject-dependent alternative relation called **ASS** (of type $\boldsymbol{e(s(st))}$) underlying the extension of the attitude verb **behaupten** [≈ claim]:

(40) a. $|\textbf{behauptet}| = (\boldsymbol{\lambda p^{st}.(\lambda x^{e}.(\forall j)[\mathbf{ASS}(x)(i)(j) \rightarrow p_j])})$
b. $\|\mathbf{ASS}\| = \lambda x.\lambda s_0.\lambda s_1.\, \vdash s_1$ is an assertoric alternative for x in s_0 $\dashv$

Here an assertoric alternative is a situation that is in line with a claim made by the subject (in the situation at hand); the details of the characterisation of **ASS** are supplied later, as part of an exercise. As in the case of $\mathbf{EVI}^{*}$ the notation of $\mathbf{ASS}^{*}$ is supposed to indicate that the missing subject is understood. Accordingly, the assertoric alternatives mentioned in (39b) are the assertoric alternatives of a person not further specified.[18] Hence the proposition expressed by (37b) consists of the situations in which this person claims (or has recently claimed) that *NN* is a conman. By the Hintikka

[17] . . . or *accessibility relation*: cf. fn. 33 in Chapter 5. – Again, we simplifyingly assume that only the word **angeblich** [≈ alleged] makes a semantic contribution to the matrix clause.

[18] The relation between **ASS** and $\mathbf{ASS}^{*}$ can be accounted for more thoroughly by the methods introduced in Chapter 8 [to be written].

Semantics of (subjectless) attitudes, (37b)can now be represented in two-sorted type logic as follows:

(41) $(\forall \boldsymbol{j})[\mathbf{ASS}^*_{\boldsymbol{i}} \rightarrow \boldsymbol{H_j}(\mathbf{n})]$

where we have assumed that $|\mathbf{Hochstapler}| = \mathbf{H_i}$ and $|NN| = \mathbf{n}$. In view of the assumed intensional identity, (41) should then also be equivalent to the type-logical translation of (37a). This observation immediately leads to an analysis of the complex noun **angeblicher Hochstapler** [≈ alleged conman] whose extension – like that of the predicate **ist ein angeblicher Hochstapler** [≈ is an alleged conman] – obviously consists of the individuals x for which it holds that x is an alleged conman:

(42) $|\mathbf{angeblicher\ Hochstapler}|$
$\equiv$ $|\mathbf{ist\ ein\ angeblicher\ Hochstapler}|$
$\equiv$ $(\boldsymbol{\lambda x^e.}(\forall \boldsymbol{j})[\mathbf{ASS}^*_{\boldsymbol{i}} \rightarrow \boldsymbol{H_j}(\boldsymbol{x})])$

Now, in order to compositionally derive the extension of the noun, it must be composed of the translation of **Hochstapler** [≈ conman] and the analysis of the adjective **angeblich** [≈ alleged] given in (38b). The following chain of type-logical equivalences shows the way:

(43) $(\boldsymbol{\lambda x^e.}(\forall \boldsymbol{j})[\mathbf{ASS}^*_{\boldsymbol{i}} \rightarrow \boldsymbol{H_j}(\boldsymbol{x})])$ *eigen*-conversion
$\equiv$ $(\boldsymbol{\lambda x.}(\forall \boldsymbol{j})[\mathbf{ASS}^*_{\boldsymbol{i}} \rightarrow (\boldsymbol{\lambda j.H_j}(\boldsymbol{x}))(\boldsymbol{j})])$ $\boldsymbol{\lambda}$-conversion
$\equiv$ $(\boldsymbol{\lambda x.}(\boldsymbol{\lambda p^{st}.}(\forall \boldsymbol{j})[\mathbf{ASS}^*_{\boldsymbol{i}} \rightarrow \boldsymbol{p}(\boldsymbol{j})])(\boldsymbol{\lambda j.H_j}(\boldsymbol{x})))$ bound renaming
$\equiv$ $(\boldsymbol{\lambda x.}(\boldsymbol{\lambda p.}(\forall \boldsymbol{j})[\mathbf{ASS}^*_{\boldsymbol{i}} \rightarrow \boldsymbol{p}(\boldsymbol{j})])(\boldsymbol{\lambda i.H_i}(\boldsymbol{x})))$
$\equiv$ $(\boldsymbol{\lambda x.}|\mathbf{angeblich}|(\boldsymbol{\lambda i.}|\mathbf{Hochstapler}|(\boldsymbol{x})))$

In the first step *eigen*-conversion is used to put the formula to the right of the arrow in the form $\boldsymbol{\alpha(j)}$, where $\boldsymbol{\alpha}$ is is a formula of type $\boldsymbol{st}$ in which $\boldsymbol{j}$ does not occur (freely). The matrix of the whole formula – the part after $\boldsymbol{\lambda x.}$ – now essentially has the structure of the translation of **angeblich** [≈ alleged] given in (38b), with the twist that the position of the $\boldsymbol{\lambda}$-bound variable $\boldsymbol{p}$ is now occupied by said $\boldsymbol{\alpha}$. In the next step this $\boldsymbol{\alpha}$ gets abstracted from by replacing it by a$\boldsymbol{\lambda}$-bound variable and applying this $\boldsymbol{\lambda}$-term to $\boldsymbol{\alpha}$ again; as a result the translation of the adjective becomes a part of the formula. (It should be noted that this abstraction process must happen with the scope of $\boldsymbol{\lambda x.}$, since the variable $\boldsymbol{x}$ is free in $\boldsymbol{\alpha}$!) The bound renaming in the final step only serves to bring in the translation of the noun **Hochstapler** [≈ conman], which contains the variable $\boldsymbol{i}$, after all.

It is immediate from the final formula in (43) how the translations of the adjective an the modified noun combine. Generalising from this case, we then get to the following interpretation of the construction, which turns out to be intensional, due to the introduction of the $\boldsymbol{\lambda i.}$:

(44) *Indirect Interpretation of Attaching Simple Adverbial Adjectives*
If N is a (complex) noun consisting of an adverbial adjective A and a (possibly complex) noun N', then:
$|N| = \boldsymbol{(\lambda x^{e}.|A|(\lambda i.|N'|(x)))}$.

Adjectives like **angeblich** [≈ alleged] are called *adverbial* (here) because their function is that of sentence adverbs, which we will briefly address in Section 6.5 [to be written]. Roughly speaking, sentence adverbs can make a claim about the proposition expressed by the whole (remaining) sentence. In the case of **angeblich** [≈ alleged] we have seen this by way of the paraphrase (37b), which in a way shows that **angeblich** [≈ alleged] there relates to the predication *NN* **ist ein Hochstapler** [≈ *NN* is a conman]. Apart from **angeblich** [≈ alleged] there are a number of further adjectives that semantically behave like sentence adverbs, as the following (approximate) paraphrase pairs are meant to indicate:

(45) a. **Hans ist ein denkbarer Kandidat.**
[≈ Hans is a conceivable candidate.]
b. **Es ist denkbar, dass Hans ein Kandidat wäre.**
[≈ It is conceivable that Hans be a candidate.]

(46) a. **Aus dem Nebenzimmer war ein gelegentliches Räuspern zu hören.**
[≈ An occasional hem could be heard from the next room.]
b. **Gelegentlich war aus dem Nebenzimmer ein Räuspern zu hören.**
[≈ Occasionally a hem could be heard from the next room.]

(47) a. **Die Parteispitze traf sich an einem geheimen Ort.**
[≈ The top-ranking party members met at a secret place.]
b. **Es war geheim, an welchem Ort sich die Parteispitze traf.**
[≈ It was kept secret at which place the top-ranking party members met.]

Not all of them – this, too, will be shown in an exercise – can be accounted for by the construction meaning given in (44), though; thence the qualification *simple*.

In the case of **angeblich** [≈ alleged] we had seen that the extension of a noun modified by an adverbial adjective does not have to be a subset of the extension of the (unmodified) noun itself: an alleged conman is not necessarily a conman. Intersective and (extensionally or intensionally) subsective adjectives are different in this respect: if they modify a noun, the resulting extension is always a subset of the extension of the bare noun: a Turkish female course participant is necessarily a female course participant; a young

lawyer must in particular be a lawyer; and a gifted pianist always needs to be a pianist. The above interpretation of the three adjective classes mentioned accounts for this fact in that the adjectives A interpreted according to it always satisfy:

(48) $\downarrow|||A\ N|||^g \subseteq \downarrow|||N|||^g$

where N is a noun. (Exercise!) However, the adverbial adjectives are not the only ones for which (48) is inadequate. As a case in point, something in the extension of **halbes Hähnchen** [≈ half a chicken] is not in the extension of **Hähnchen** [≈ chicken]; still a sentence like (49a) can hardly be paraphrased along the lines of (49b), where X is some attitude corresponding to the adjective **halb** [≈ half a]:

(49) a. **Fritz kauft ein halbes Hähnchen.**
[≈ Fritz is buying half a chicken.]
b. **Es ist X, dass Fritz ein Hähnchen kauft.**
[≈ It is X that Fritz is buying a chicken.]

Instead **halb** [≈ half a] seems to modify its head noun N in a way so that the extension consists of halves of objects that are in the extension of N. The following interpretation captures this idea:

(50) a. $|\mathbf{halb}| = \mathbf{H_i}$, where $\mathbf{H}$ is a constant of type $\boldsymbol{s((et)(et))}$.
b. $\|\mathbf{H}\| = \lambda s.\lambda P.\lambda x.\ \vdash$ there is a y such that y is as big as x and $P(x \oplus y) = 1\ \dashv$

Here '$x \oplus y$' designates the *fusion* of the objects x and y, which is an object that consists of the parts x and y and has no further parts (= parts that do not overlap with x or y).[19] The idea behind (50) is that the application of the extension of **halb** [≈ half a] to that of **Hähnchen** [≈ chicken] is (the characteristic function of) the set of objects that together with another object of the same size make up for a chicken. At first blush, this characterisation of the intension of **halb** [≈ half a] may appear plausible; however, we will see in a second that it is not correct. Before this we first notice that although, according to (50), **halb** [≈ half a] has the (extension) type of **jung** [≈ young], but still is not extensionally subsective – since half a chicken is not a chicken after all. In other words, modification by **halb** [≈ half a] must

[19] The notation '$\oplus$' originates from *mereology* (the logic of the part-whole relation); of course, it has nothing to do with the ad-hoc use of the same symbol in Section 2.2. – The notion of size in (50b) depends on the object under scrutiny: in the case of a chicken, weight and volume are decisive, for a circle it is its perimeter, for a time span its length, etc. The last example also shows that the complementing second half needs not be uniquely determined: the time x between 6pm and 7pm is half an hour but there are two distinct y of the same length that complement x to an element $x \oplus y$ of the extension of **Stunde** [≈ hour], viz., the immediately preceding half hour and the one that immediately follows.

not be interpreted in terms of (30); for otherwise the effect described in (48) would occur. Instead, the extension of the adjective must be applied to that of the noun without intersecting at the same time:

(51) *Indirect Interpretation of Attaching Extensionally Modifying Adjectives*
If N is a (complex) noun consisting of an extensionally modifying adjective A and a (possibly complex) noun N', then:
$|N| = |A|(|N'|)$.

(51) must not be read as an addition to the above translation rules, but may instead replace (30) – provided the lexical interpretations of the extensionally subsecting adjectives are adapted. Though an interpretation of **junger Politiker** [≈ young politician] according to (51) and on the basis of the indirect interpretation (31) of **jung** [≈ young] will not lead to the desired result (as will be shown in an exercise), the following revision of (31a) will:

(52) a. $|\mathbf{jung}| = (\boldsymbol{\lambda P^{et}.\lambda x^{e}.[P(x) \wedge \mathbf{J}_i(P)(x)]})$,
where **J** is a constant of type ***s((et)(et))***.

b. $\|\mathbf{J}\| = \lambda s.\lambda M.\lambda x. \vdash \overline{\alpha_s^M} \gg \alpha_s(x) \dashv$, where $\alpha_s(x)$ is the age of the individual x in the situation s and $\overline{\alpha_s^M}$ is the average value of the age function α_s over the elements of the set M.

(It should be noted that the interpretation of the constant **J** has not changed: (31b) and (52b) are identical!) Corresponding revisions of the lexical meanings can obviously be carried out for all subsective adjectives, which are then construed as extensionally modifying and thus subject to (51). In this sense (51) can be applied to all extensionally subsective adjectives, but the rule actually fails on the example originally motivating it: despite appearances – and the interpretation given in (50) – **halb** [≈ half a] is not extensional after all. For even if all bread rolls in a bakery (situation) were poppy seed bread rolls and thus the extension of **Brötchen** [≈ bread roll] and **Mohnbrötchen** [≈ poppy seed bread roll] coincided, not all bread roll halves would have to be poppy seed bread roll halves – maybe there is still half a sesame seed bread roll lying around somewhere. The example also shows a fundamental flaw in the above analysis (50) of **halb** [≈ half a]. For there we had taken it for granted that for each half of an object a complementing counterpart always exists, or had existed. But where the building of a bridge was unfinished, half a bridge may have me into existence without there ever being or having been the other half. The half-built bridge is merely an object half of which – a matching piece of equal size – is missing to be a bridge. Rather than investigating the details of the lexico-semantic analysis of **halb** [≈ half a], we will leave it at that and merely state that it cannot be an extensionally modifying adjective but instead (apparently) must be interpreted as intensionally modifying, with extension type ***(s(et))(et)*** but without being

intensionally subsective. For the modification by adjectives of this type a construction meaning parallel to (51) can obviously be assumed:

(53) *Indirect Interpretation of Attaching Intensionally Modifying Adjectives*
IF N is a (complex) noun consisting of an intensionally modifying adjective A and a (possibly complex) noun N', then:
$|N| = |A|(\boldsymbol{\lambda i.}|N'|)$.

As in the extensional case, the lexical analyses of all intensionally subsective adjectives may now be modified so as to be covered by (53); an exercise will illustrate this with the example **begabt** [≈ gifted].

Let us take stock. From a semantic point of view, adjectives fall into different classes, at least some of which we have encountered here:

- *Intersective* adjectives have extensions of type ***et*** and combined with their head nouns like restrictive relative clauses.
- *Extensionally modifying* adjectives are combined by functional application; their extension type is accordingly ***(et)(et)***. *Extensionally subsective* adjectives, whose application narrows down the extension of the head noun, may be construed as a special case of the extensionally modifying adjectives.
- In the case of *intensionally modifying* adjectives the extension is applied to the intension of the head noun; their extension type is accordingly ***(s(et))(et)***. *Intensionally modifying* adjectives, whose application narrows down the extension of the head noun, may again be construed as a special case.
- *Adverbial* adjectives have the extension type ***(st)t*** of sentence adverbs; in *simple* cases they directly combine with their head noun and apply to the proposition expressed by predication.

In the next section we will see how this heterogeneity in the type assignment can be avoided for the price of a certain abstractness. The techniques to be introduced there are quite general and can also be used outside the semantics of adjectives. However, before leaving this area, it should be mentioned that we have left quite a few semantic phenomena out of account. In particular, apart from the *prenominal* (or *attributive*) position considered here, most adjectives can also be used *predicatively*:

(54) a. ?**Selin ist türkisch.**
[≈ Selin is Turkish.]
b. **Der Anwalt war jung.**
[≈ The lawyer was young.]

c. *__Mancher Hochstapler ist angeblich.__
[≈ *Some conmen are alleged.]
d. __Fritz ist sehr begabt.__
[≈ Fritz is very gifted.]
e. __Der Treffpunkt war geheim.__
[≈ The meeting place was secret.]

We will not go into this construction because it is not a modification. In a similar vein, we will ignore a number of further phenomena that are characteristic for (some) adjectives – like comparison and antonymy. The ambiguities in (55) and the acceptability contrasts in (56) are just supposed to substantiate that these areas, too, are full of semantic secrets:

(55) a. __Fritz kennt bessere Schachspieler als Hans.__
[≈ Fritz knows better chess players than Hans.]
b. __Hans läuft schneller als Fritz denkt.__
[≈ Hans runs faster that Fritz thinks.]
c. __Meier hat die meisten Treffer erzielt.__
[≈ Meier achieved (the) most hits.]

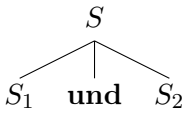
(56) a. __Das Haus ist 10 Meter höher als breit.__
[≈ The house is 10 meters higher than broad.]
b. ?__Das Haus ist 10 Meter schmaler als hoch.__
[≈ ?The house is 10 meters narrower than high.]
c. __Das Haus ist 20 Meter hoch.__
[≈ ?The house is 20 meters high.]
d. ??__Das Haus ist 10 Meter schmal.__
[≈ ??The house is 10 meters narrow.]

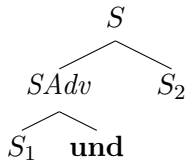
6.3 Type Shifts

Given the differences in their semantic behaviour, we assigned different extension types to different adjectives. As a consequence of this semantic differentiation, we distinguish between (at least) four different syntactic constructions of adjectival modification, which accordingly are interpreted in different ways. In this respect the situation does not fundamentally differ from the interpretation of subject or object attachment, where we also distinguished between predication, quantification, and raising or opacity. In each of these cases we had assumed that the syntactic structure somehow codes – by suitable features maybe – which subcase is at stake. In the current section we will get acquainted with several methods of *flexible interpretation* that allow for making the semantic differentiations mentioned without syntactic help.

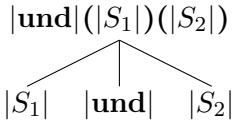
We had seen in connection with (29) that it is not possible to interpret **jung**

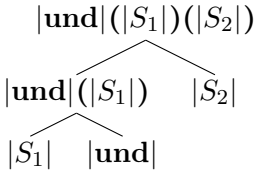
[≈ young] as an intersective adjective and instead assigned to it the extension type ***(et)(et)***. From this it does not follow, though, that the extension of **jung** [≈ young] and that of an intersective adjective like **türkisch** [≈ Turkish] cannot be of the same type. For although this common type cannot be ***et***, it is quite possible to analyse the extensions of both adjectives as objects of type ***(et)(et)***. To see how this works, we first recall from Section 5.1 the technique of replacing simultaneous application of a binary function to successive application of two unary functions. This trick, better known as *Currying* allowed us to construe the extension of **und** [≈ and] as a function of type ***t(tt)***. Even though we had been assuming a ternary coordinate structure as in (57a) we could just as well have transferred the binary and successive procedure to syntax, e.g., by using the bracketing in(57b):

(57) a. 

b. 

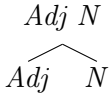
With the alternative bracketing (57b) the left (main) node serves as a kind of sentence adverb that modifies the second conjunct. The compositional indirect interpretation of (57b) is as obvious as before:

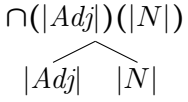
(58) a. 

b. 

In (58b) the first conjunct and the conjunction **und** [≈ and] together form an expression of their own, whose extension is applied to the truth value of the second conjunct. As one can easily see, the extension of this constituent (classified as *SAdv*) must then be of type ***tt***.

Though modification by intersective adjectives is not a ternary construction, it is interpreted by simultaneous application of a binary function, intersection, which for reasons of transparency we will abbreviate by the corresponding settheoretic symbol:

(59) a. 

b. 

In (59b) ∩ is a constant of type ***(et)((et)(et))*** denoting the operation on characteristic functions that corresponds to set intersection:[20]

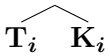
[20]For the sake of accuracy, it should be mentioned that the symbol '∩' is used to indicate

(60) $\|\cap\| = \lambda P.\lambda Q.\lambda x. \vdash P(x) = Q(x) = 1 \dashv$

With this notation the above example **türkisch** [≈ Turkish] comes out as follows:

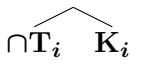
(61) a. **türkische Kursteilnehmerin**

b. $\cap(\mathbf{T}_i)(\mathbf{K}_i)$



where **T** and **K** are the constants of type ***s(et))*** interpreted in (24). As in the transition from a. to b. in (58), we can now compress the extension of the left consituent in a single binary operation – intersection, in the case at hand:

(62) $\cap(\mathbf{T}_i)(\mathbf{K}_i)$



Unlike the one (58b), the left daughter node in (62) does not branch; for it is not the result of a combination of two expressions. Instead, we may construe it as the translation of the adjective **türkisch** [≈ Turkish] itself. The above translation (25a) would then have to be replaced by:[21]

(25a) $|\mathbf{türkisch}_{\cap}| = \mathbf{T}_i$

(63) $|\mathbf{türkisch}_{ext}| = \cap(\mathbf{T}_i) \equiv (\boldsymbol{\lambda P^{et}.(\lambda x^{e}.[\mathbf{T}_i(x) \wedge P(x)])})$

It is readily seen that according to (63), the adjective **türkisch** [≈ Turkish] has the extension type ***(et)(et)***. It can then be classified as extensionally modifying without affecting the result of meaning composition. Of course, this was the very idea behind passing from (61a) to (62):

(64)	$\|\mathbf{türkische\ Kursteilnehmerin}\|$	by (26)
=	$(\boldsymbol{\lambda x^{e}.}[\|\mathbf{Kursteilnehmerin}\|(x) \wedge \|\mathbf{türkisch}_{\cap}\|(x)])$	by (24)
=	$(\boldsymbol{\lambda x.}[\mathbf{K}_i(x) \wedge \mathbf{T}_i(x)])$	$\boldsymbol{\lambda}$-conversion
≡	$(\boldsymbol{\lambda P^{et}.(\lambda x.}[P(x) \wedge \mathbf{T}_i(x)]))(\mathbf{K}_i)$	$\boldsymbol{\lambda}$-conversion
≡	$(\boldsymbol{\lambda Q^{et}.(\lambda P.(\lambda x.}[P(x) \wedge Q(x)])))(\mathbf{T}_i)(\mathbf{K}_i)$	exercise
≡	$\cap(\mathbf{T}_i)(\mathbf{K}_i)$	by (63)
≡	$\|\mathbf{türkisch}_{ext}\|(\mathbf{K}_i)$	by 5.5, (74)
≡	$\|\mathbf{türkisch}_{ext}\|(\|\mathbf{Kursteilnehmerin}\|)$	by (51)
≡	$\|\mathbf{türkisch}_{ext}\ \mathbf{Kursteilnehmerin}\|$	

What we did using **türkisch** [≈ Turkish] as an example, can principally be

arbitrary intersections and, in particular, does not stand for a set-theoretic *function*.

[21] We distinguish the two variants of indirectly interpreting the surface form **türkisch** [≈ Turkish] by corresponding subscripts on the underlying forms, in order to be able to apply the same translation function to them – and not to postulate an ambiguity.

done with all intersective adjectives: if they get assigned type-logical translations *à la* (63), they can all be subsumed under the extensionally modifying adjectives, thus proving redundant the assumed specific modification construction (26) for intersective adjectives.

The process of unifying the constructions considered from the previous section does not have to stop here. We had already seen that extensionally subsective adjectives like **jung** [≈ young] could be interpreted as extensionally modifying once pertinent arrangements have been made in their lexical analysis; in a similar vein, it was shown (in an exercise) that intensionally subsective adjectives like **begabt** [≈ gifted] can be interpreted as intensionally modified. But even the modifying adjectives *in toto* may be reduced to a common denominator. For although an intensional adjective like **begabt** [≈ gifted] cannot be reduced to a function of type $\boldsymbol{(et)(et)}$(as an exercise will prove in detail), one can reversely assign extensions of type $\boldsymbol{(s(et))(et)}$ to adjectives like **jung** [≈ young] (or **türkisch** [≈ Turkish]) without changing the result of modification. In a situation s, these 'new' extensions only need to operate on the intension instead of the extension of the noun they modify; but given such s, the intension fully determines the extension. An extensionally modifying adjective A_{ext} may thus be re-interpreted as intensionally modifying:

(65) $\quad |A_{int}| = \boldsymbol{(\lambda\Phi^{s(et)}.|A_{ext}|(\Phi_i))}$

As a case in point, we obtain the following interpretation of nominal modification by **jung** [≈ young]:

(66)	$\|\mathbf{junger}_{int}\ \mathbf{Politiker}\|$	by (53)
=	$\|\mathbf{jung}_{int}\|(\boldsymbol{\lambda i.}\|\mathbf{Politiker}\|)$	**P** of type $\boldsymbol{s(et)}$
=	$\|\mathbf{jung}_{int}\|(\boldsymbol{\lambda i.}\mathbf{P}_i)$	η-conversion, cf. (73e) in Section 5.4
≡	$\|\mathbf{jung}_{int}\|(\mathbf{P})$	by (65)
≡	$(\boldsymbol{\lambda\Phi^{s(et)}.}\|\mathbf{jung}_{ext}\|(\Phi_i))(\mathbf{P})$	$\boldsymbol{\lambda}$-conversion
≡	$\|\mathbf{jung}_{ext}\|(\mathbf{P}_i)$	see above
≡	$\|\mathbf{jung}_{ext}\|(\|\mathbf{Politiker}\|)$	by (51)
≡	$\|\mathbf{junger}_{ext}\ \mathbf{Politiker}\|$	

And, of course, (65) can also be applied to intersective adjectives after they have been re-classified as in (63):

(67)	$\|\mathbf{türkische}_{int}\ \mathbf{Kursteilnehmerin}\|$	by (53)
=	$\|\mathbf{türkisch}_{int}\|(\boldsymbol{\lambda i.}\|\mathbf{Kursteilnehmerin}\|)$	by (24)
=	$\|\mathbf{türkisch}_{int}\|(\boldsymbol{\lambda i.}\mathbf{K}_i)$	η-conversion, cf. (73e) in Section 5.4
≡	$\|\mathbf{türkisch}_{int}\|(\mathbf{K})$	by (66)
≡	$(\boldsymbol{\lambda\Phi^{s(et)}.}\|\mathbf{türkisch}_{ext}\|(\Phi_i))(\mathbf{K})$	$\boldsymbol{\lambda}$-Konversion
≡	$\|\mathbf{türkisch}_{ext}\|(\mathbf{K}_i)$	by (64)
≡	$\|\mathbf{türkische}_{\cap}\ \mathbf{Kursteilnehmerin}\|$	

Now that we have seen that the intersective and the extensionally modifying adjectives can all be treated as special cases of the intensionally modifying ones, the adverbial adjectives form the only remaining exceptional case among the classes considered in the previous section. But even they can be reduced to the common denominator of intensional modification. To see how this works, let us recall (44), which translates modifications of a noun N by an adverbial adjective A_{adv} into type logic:

(68) $\quad |A_{adv}\ N| = (\boldsymbol{\lambda x^e.}|A_{adv}|(\boldsymbol{\lambda i.}|N|(\boldsymbol{x})))$

In order to obtain the same result by applying an intensionally modifying version of A_{adv} to the intension of N, the latter must merely be put in argument position:

(69)	$(\boldsymbol{\lambda x^e.}|A_{adv}|(\boldsymbol{\lambda i.}|N|(\boldsymbol{x})))$	*eigen*-conversion
$\equiv$	$(\boldsymbol{\lambda x.}|A_{adv}|(\boldsymbol{\lambda i.}(\boldsymbol{\lambda i.}|N|)(\boldsymbol{i})(\boldsymbol{x})))$	$\boldsymbol{\lambda}$-conversion
$\equiv$	$(\boldsymbol{\lambda \Phi^{s(et)}.}(\boldsymbol{\lambda x.}|A_{adv}|(\boldsymbol{\lambda i.\Phi}(\boldsymbol{i})(\boldsymbol{x}))))\ (\boldsymbol{\lambda i.}|N|)$	

This is not the first time that we exploit the idea behind the equivalent reformulation in (69):[22] in order to represent the entire formula as a result of applying a function to an intension – in this case that of N, the latter must first be formed from the translation of the extension using *eigen*-conversion, whereupon the result can be $\boldsymbol{\lambda}$-abstracted. The remaining function then has precisely the type $\boldsymbol{(s(et))(et)}$ of the intensional adjective and can thus be used for re-categorising A_{adv}:

(70) $\quad |A_{int}| = (\boldsymbol{\lambda \Phi^{s(et)}.}(\boldsymbol{\lambda x^e.}|A_{adv}|(\boldsymbol{\lambda i.\Phi}(\boldsymbol{i})(\boldsymbol{x}))))$

The correctness proof of (70) in the style of (66) and (67) is left to an exercise this time.

The unification of adjectival semantics presented above is a special case of an interpretive strategy that runs by the name of *generalising to the worst case*.[23] According to it, (if possible) all expressions of a given syntactic category receive extensions of the same type, viz., the *smallest* type in which all extensions of expressions of this category are *representable*. Here, representation must be understood as embedding, i.e., a function definable in purely logical terms that maps all objects of one ('smaller') type to objects of another ('larger') type without ever mapping distinct objects to the same one, and doing so in a manner as natural as possible. Such mappings are known as *type shifts* in semantics.[24] The re-interpretations used for re-categorising

[22] The first time was in determining the extension of the rasing verb **scheinen** [≈ seem]; cf. (133) in Section 5.7.

[23] This self-explanatory term appears to have originated with Barbara Partee and describes the spirit behind the unified type assignments in Richard Montague's *The Proper Treatment of Quantification in Ordinary English* (1973).

[24] A comprehensive systematic account of the theory of type shifting can be found in Johan

various adjectives can thus be seen as type shifts from a 'smaller' input type a to a 'larger' output type a^*:

(71)

Input type	*Output type*	*Shift*	
$\boldsymbol{et}$	$\boldsymbol{(et)(et)}$	$(\boldsymbol{\lambda Q^{et}.(\lambda P^{et}.(\lambda x^{e}.[P(x) \wedge Q(x)]))})$	cf. (63)
$\boldsymbol{(et)(et)}$	$\boldsymbol{(s(et))(et)}$	$(\boldsymbol{\lambda M^{(et)(et)}.(\lambda \Phi^{s(et)}.M(\Phi(i)))})$	cf. (65)
$\boldsymbol{(st)t}$	$\boldsymbol{(s(et))(et)}$	$(\boldsymbol{\lambda O^{(st)t}.(\lambda \Phi^{s(et)}.(\lambda x^{e}.O(\lambda i.\Phi(i)(x))))})$	cf. (70)

(71) is to be understood as saying that by applying the type shift given, an arbitrary object of input type a is represented as an object of output type a^*. We have written the type shifts themselves as type-logical formulae to ensure that they are functions of type (aa^*) that are definable by purely (type-) logical means. If, e.g., the formula given in the first line is applied to the translation of the intersective version of **türkisch** [≈ Turkish][25], the result is the interpretation of the adjective as extensionally modifying as given in (63):

(72)
$$\begin{array}{lll} & (\boldsymbol{\lambda Q^{et}.(\lambda P^{et}.(\lambda x^{e}.[P(x) \wedge Q(x)]))})\ (|\mathbf{türkisch}_{\cap}|) & \text{by (25a)} \\ = & (\boldsymbol{\lambda Q.(\lambda P.(\lambda x.[P(x) \wedge Q(x)]))})\ (\mathbf{T}_i) & \boldsymbol{\lambda}\text{-conversion} \\ \equiv & (\boldsymbol{\lambda P.(\lambda x.[P(x) \wedge \mathbf{T}_i(x)])}) & \text{cf. (63)} \\ \equiv & |\mathbf{türkisch}_{ext}| & \end{array}$$

Given the result achieved in (72), the second line of (71) may then apply in the same way to shift from $\boldsymbol{(et)(et)}$ to $\boldsymbol{(s(et))(et)}$ – and with the known result expounded in(67). A further type shift ensues by plugging the two processes together:

(71$^+$)

Input type	*Output type*	*Shift*
$\boldsymbol{et}$	$\boldsymbol{(s(et))(et)}$	$(\boldsymbol{\lambda P^{et}.(\lambda \Phi^{s(et)}.(\lambda x^{e}.[P(x) \wedge \Phi(i)(x)]))})$

Adjectival nominal modification is not the only place where type shifts may be employed. Thus, e.g., the shift (63) from $\boldsymbol{et}$ to $\boldsymbol{(et)(et)}$ may equally be applied to (restrictive) relative clauses – thereby bringing them closer to the unified adjectives:

(73)
$$\begin{array}{lll} & (\boldsymbol{\lambda P^{et}.(\lambda \Phi^{s(et)}.(\lambda x^{e}.[P(x) \wedge \Phi(i)(x)]))}) & \\ & (|\mathbf{die\ in\ der\ zweiten\ Reihe\ sitzt}|) & \text{by (11a)} \\ \equiv & (\boldsymbol{\lambda P.(\lambda \Phi.(\lambda x.[P(x) \wedge \Phi(i)(x)]))})\ (\mathbf{Z}_i) & \boldsymbol{\lambda}\text{-conversion} \\ \equiv & (\boldsymbol{\lambda \Phi.(\lambda x.[\mathbf{Z}_i(x) \wedge \Phi(i)(x)])}) & \end{array}$$

Thus type-shifted, the relative clause and then be attached to the noun in

van Benthem's bookh *Language in Action* (1991).

[25] Strictly speaking, it is not the type-logical formulae themselves that are applied to each other but their semantic values. We ignore such object-meta-language subtleties in the interest of perspicuity.

analogy to intensionally modifying adjectives, thus replacing the above rule (9):

(9) *Indirect Interpretation of the Attachment of Restrictive Relative Clauses*
If N is a (complex) noun consisting of a (possibly complex) noun N' and a relative clause M, then:
$|N| = (\boldsymbol{\lambda x^e.}[|N'|(\boldsymbol{x}) \wedge |M|(\boldsymbol{x})])$.

(74) *Indirect Interpretation of Attaching of Type-Shifted Relative Clauses*
If N is a (complex) noun consisting of a (possibly complex) noun N' and a type-shifted relative clause M^+, then:
$|N| = |M^+|(\boldsymbol{\lambda i.}|N'|)$.

The final result is, of course, equivalent again to the original result of intersection – but it comes about on the basis of a unified semantics of modification:

(75)	$\|\textbf{Kursteilnehmerin, die in der zweiten Reihe sitzt}^+\|$	by (74)
=	$\|\textbf{die in der zweiten Reihe sitzt}^+\|\ (\boldsymbol{\lambda i.}\|\textbf{Kursteilnehmerin}\|)$	see above
≡	$\|\textbf{die in der zweiten Reihe sitzt}^+\|\ (\boldsymbol{\lambda i.}\mathbf{K}_i)$	η-conversion
≡	$\|\textbf{die in der zweiten Reihe sitzt}^+\|\ (\mathbf{K})$	by (73)
≡	$(\boldsymbol{\lambda\Phi^{s(et)}.(\lambda x^e.}[\mathbf{Z}_i(\boldsymbol{x}) \wedge \boldsymbol{\Phi(i)(x)}]))\ (\mathbf{K})$	$\boldsymbol{\lambda}$-conversion
≡	$(\boldsymbol{\lambda x.}[\mathbf{Z}_i(\boldsymbol{x}) \wedge \mathbf{K}_i(\boldsymbol{x})])$	propositional logic
≡	$(\boldsymbol{\lambda x.}[\mathbf{K}_i(\boldsymbol{x}) \wedge \mathbf{Z}_i(\boldsymbol{x})])$	cf. (8)
≡	$\|\textbf{Kursteilnehmerin, die in der zweiten Reihe sitzt}\|$	

The unification of the semantics of adjectival modification and modification by (and attaching) restrictive relative clauses is somewhat dubious of course inasmuch the latter are assigned obviously unnecessarily complex semantic values. We will return to questions like this one towards the end of the section, after taking a look at further fields of application for type shifts.

We have come across several extension types within the same category before. In particular, we assumed a fundamental distinction between proper names and quantifying noun phrases that is reflected in the semantic type as well as in meaning composition. Yet as in the case of adjectives, in this case too, a common denominator type can be found for the two syntactic categories. Here the quantifying noun phrases with their visibly more complex type form the worst case. In otder to represent the extensions of proper names – i.e., the name bearers of type $\boldsymbol{e}$ – bearers of type $\boldsymbol{(et)t}$ of quantifiers, we must only perform the corresponding subtractions on sentence and predicate etensions. The extensions of type shifted proper names N^* then come out as functions that assign to the extensions of arbitrary predicates P the respective truth values of the sentences N *VP*, which in turn result

from functional application:[26]

(71*)

Input type	*Output type*	*Shift*
$\boldsymbol{e}$	$\boldsymbol{(et)t}$	$\boldsymbol{(\lambda x^{e}.(\lambda P^{et}.P(x)))}$

Obviously, the proper names type-shifted by (71*) not only have the extension type of quantified noun phrases; the can also be interpreted like the latter in the compositional process, as the following example illustrates, where the typeshifted names have been marked by an asterisk:

(76)
$$
\begin{array}{lll}
 & |\mathbf{Eike}^{*}\ \mathbf{trifft}\ \mathbf{Fritz}^{*}| & \text{by 5.5, (88), } \textit{quantifier} \text{ in subject position!} \\
= & |\mathbf{Eike}^{*}|(|\mathbf{trifft}\ \mathbf{Fritz}^{*}|) & \text{by 5.5, (89), } \textit{quantifier} \text{ in object position!} \\
= & |\mathbf{Eike}^{*}|(\boldsymbol{\lambda x^{e}}.|\mathbf{Fritz}^{*}|(\boldsymbol{\lambda y^{e}}.|\mathbf{trifft}|(\boldsymbol{y})(\boldsymbol{x}))) & \text{by (71*)} \\
= & (\boldsymbol{\lambda x^{e}}.(\boldsymbol{\lambda P^{et}}.\boldsymbol{P}(\boldsymbol{x})))(|\mathbf{Eike}|)(\boldsymbol{\lambda x}. & \\
 & (\boldsymbol{\lambda x^{e}}.(\boldsymbol{\lambda P^{et}}.\boldsymbol{P}(\boldsymbol{x})))(|\mathbf{Fritz}|)(\boldsymbol{\lambda y}.|\mathbf{trifft}|(\boldsymbol{y})(\boldsymbol{x}))) & \text{by 5.5, (74)} \\
= & (\boldsymbol{\lambda x}.(\boldsymbol{\lambda P}.\boldsymbol{P}(\boldsymbol{x})))(\mathbf{e})(\boldsymbol{\lambda x}.(\boldsymbol{\lambda x}.(\boldsymbol{\lambda P}.\boldsymbol{P}(\boldsymbol{x})))(\mathbf{f})(\boldsymbol{\lambda y}.|\mathbf{trifft}|(\boldsymbol{y})(\boldsymbol{x}))) & \\
\equiv & \ldots & \text{exercise} \\
\equiv & \mathbf{T}_{\boldsymbol{i}}(\mathbf{f})(\mathbf{e}) & \text{cf. 5.2, (15)} \\
\equiv & |\mathbf{Eike}\ \mathbf{trifft}\ \mathbf{Fritz}| &
\end{array}
$$

One can see from (76) how the type-shifted names act as quantifiers in subject and object positions. The shift (71*) can also be applied to those constructions that we had reserved to quantifying nouns in the preceding chapter, viz., the subject position of raising verbs and the object position of opaque verbs; this will be done in an exercise.[27] In the meantime we turn to the opaque verbs themselves, for which we established the extension type $\boldsymbol{(s((et)t))(et)}$ in Section 4.6. Since **suchen** & Co. [≈ seek & its ilk] morpho-syntactically behave like ordinary transitive verbs, the question arises whether their type $\boldsymbol{e(et)}$ could also be shifted to the obviously more complex type of opaque verbs. This is precisely what the following type shift, which had come across before in disguise[28] and which we will not bother to derive here does precisely this:

(71′*)

Input type	*Output type*	*Shift*
$\boldsymbol{e(et)}$	$\boldsymbol{(s((et)t))(et)}$	$\boldsymbol{(\lambda R^{e(et)}.(\lambda \wp^{s((et)t)}.(\lambda x^{e}.\wp_{i}(\lambda y^{e}.R(x,y)))))}$

The exemplary correctness proof of the type shift has become a matter of routine by now; we leave the proper names unscathed this time:

[26] The type shift defined in (71*) is also known as *Montague Lifting*. We had already encountered it in an exercise at the end of Chapter 3.

[27] We will return to the interpretation of proper names in the object position of opaque verbs in the next chapter [which still needs to be written].

[28] – to wit, in Section 3.5, (92), where one can find an extensional variant of (71′*) formulated in terms of direct interpretation.

(77) $|\textbf{Eike kennt}^{*}\ \textbf{jeden Mann}|$ by 5.5, (78): predication
$= |\textbf{kennt}^{*}\ \textbf{jeden Mann}|(|\textbf{Eike}|)$ by 5.7, (142): *opaque* verb!
$= |\textbf{kennt}^{*}|(\boldsymbol{\lambda i}.|\textbf{jeden Mann}|)(|\textbf{Eike}|)$ by 5.5, (90): quantifying noun phrase
$= |\textbf{kennt}^{*}|(\boldsymbol{\lambda i}.|\textbf{jeden}|(|\textbf{Mann}|))(|\textbf{Eike}|)$ by (71)′*
$= (\boldsymbol{\lambda R^{e(et)}}.(\boldsymbol{\lambda \wp^{s((et)t)}}.(\boldsymbol{\lambda x^{e}}.\wp_i(\boldsymbol{\lambda y^{e}}.\boldsymbol{R(x,y)})))))(|\textbf{kennt}|)$
$(\boldsymbol{\lambda i}.|\textbf{jeden}|(|\textbf{Mann}|))(|\textbf{Eike}|)$ cf. 5.5, (74) & (76d), 5.6, (104) & (107)
$= (\boldsymbol{\lambda R}.(\boldsymbol{\lambda \wp}.(\boldsymbol{\lambda x}.\wp_i(\boldsymbol{\lambda y}.\boldsymbol{R(x,y)})))))(\mathbf{K}_i)$
$(\boldsymbol{\lambda i}.(\boldsymbol{\lambda Q^{et}}.(\boldsymbol{\lambda P^{et}}.(\forall \boldsymbol{x^{e}})[\boldsymbol{Q(x)} \rightarrow \boldsymbol{P(x)}]))(\mathbf{M}_i))(\mathbf{e})$ $\boldsymbol{\lambda}$-conversion
$\equiv (\boldsymbol{\lambda \wp}.(\boldsymbol{\lambda x}.\wp_i(\boldsymbol{\lambda y}.\mathbf{K}_i(\boldsymbol{x,y}))))$
$(\boldsymbol{\lambda i}.(\boldsymbol{\lambda Q}.(\boldsymbol{\lambda P}.(\forall \boldsymbol{x})[\boldsymbol{Q(x)} \rightarrow \boldsymbol{P(x)}]))(\mathbf{M}_i))(\mathbf{e})$ $\boldsymbol{\lambda}$-conversion
$\equiv (\boldsymbol{\lambda \wp}.(\boldsymbol{\lambda x}.\wp_i(\boldsymbol{\lambda y}.\mathbf{K}_i(\boldsymbol{x,y}))))$
$(\boldsymbol{\lambda i}.(\boldsymbol{\lambda P}.(\forall \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x}) \rightarrow \boldsymbol{P(x)}]))(\mathbf{e})$ $\boldsymbol{\lambda}$-conversion
$\equiv (\boldsymbol{\lambda x}.(\boldsymbol{\lambda i}.(\boldsymbol{\lambda P}.(\forall \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x}) \rightarrow \boldsymbol{P(x)}]))(\boldsymbol{i})(\boldsymbol{\lambda y}.\mathbf{K}_i(\boldsymbol{x,y})))(\mathbf{e})$ $\boldsymbol{\lambda}$-conversion
$\equiv (\boldsymbol{\lambda i}.(\boldsymbol{\lambda P}.(\forall \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x}) \rightarrow \boldsymbol{P(x)}]))(\boldsymbol{i})(\boldsymbol{\lambda y}.\mathbf{K}_i(\mathbf{e},\boldsymbol{y}))$ *eigen*-conversion
$\equiv (\boldsymbol{\lambda P}.(\forall \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x}) \rightarrow \boldsymbol{P(x)}])(\boldsymbol{\lambda y}.\mathbf{K}_i(\mathbf{e},\boldsymbol{y}))$ $\boldsymbol{\lambda}$-conversion
$\equiv (\forall \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x}) \rightarrow (\boldsymbol{\lambda y}.\mathbf{K}_i(\mathbf{e},\boldsymbol{y}))(\boldsymbol{x})]$ $\boldsymbol{\lambda}$-conversion
$\equiv (\forall \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x}) \rightarrow \mathbf{K}_i(\mathbf{e},\boldsymbol{x})]$ routine
$\equiv \ldots$
$\equiv |\textbf{Eike kennt jeden Mann}|$

In view of such complex types, the homogeneity forced by type shifts like (71*) and, above all, (71′*) seems to be little more than a formalistic game. But uniformity can also bring along serious advantages. This will become clear if we take a look at a construction type mentioned at the beginning of this section: coordination.

6.4 Direct Coordination

Obviously, the conjunctions **und** [≈ and] and **oder** [≈ or] are not exclusively flanked by sentences but may connect expressions of any kind – as long as they belong to the same category:[29]

(78) a. **Maria raucht eine Zigarette und sieht fern.** predicates
[≈ Maria is smoking a cigarette and watching TV.]

[29]The parenthetical additions are supposed to increase idiomaticity without affecting literal meaning. We take it that **gar** [≈ at all] is an emphatic strengthening of **kein-** [≈ no] (and that, consequently, **gar kein-** [≈ no ... at all] is a complex determiner), while **entweder** [≈ either] and **oder** [≈ or] form one single morpheme that is synonymous with **oder** [≈ or] alone. Neither assumption is totally unproblematic, but in the current context they are quite harmless. – It should be noted that in (78c) **Sport** [≈ sport] is used in the sense of the sortal **Sportart** [≈ form of sport] and not as a mass noun, as in the proverbial **Sport ist Mord** [≈ Sport is murder].

b. **Alain kauft oder mietet ein Zimmer.** transitive verbs
[≈ Alain is buying or renting a room.]

c. **Die meisten Deutschen mögen [entweder] jeden oder [gar] keinen Sport.** determiners
[≈ Most Germans [either] like every or no form of sport at all.]

d. **Hans besitzt ein Moped oder ein Fahrrad.** quantifying nominals
[≈ Hans owns a moped or a bicycle.]

e. **Fritz und Eike wohnen in Tübingen.** proper names
[≈ Fritz and Eike live in Tübingen.]

The examples in (78) obviously cannot be accounted for in terms of coordination as clausal combination. However, the following paraphrases indicated that these usages are not totally different after all:

(79) a. **Maria raucht eine Zigarette, und Maria sieht fern.**
[≈ Maria is smoking a cigarette and Maria is watching TV.]

b. **Alain kauft ein Zimmer, oder Alain mietet ein Zimmer.**
[≈ Alain is buying a room or Alain is renting a room.]

c. **Für die meisten Deutschen gilt: sie mögen jeden Sport, oder sie mögen keinen Sport.**
[≈ For most Germans it holds that they like every form of sport or that they like no form of sport.]

d. **Hans besitzt ein Moped, oder Hans besitzt ein Fahrrad.**
[≈ Hans owns a moped or Hans owns a bicycle.]

e. **Fritz wohnt in Tübingen, und Eike wohnt in Tübingen.**
[≈ Fritz lives in Tübingen and Eike lives in Tübingen.]

In order to see the systematicity in passing from (78) to (79), let us first scrutinise case a. (78a) and (79a) say the same and can both be rendered by the type-logical formula (80a), where for simplicity we have assumed **R[aucht]** [≈ smokes] and **G[lotzt]** [≈ goggles] constants of type ***s(et)***:

(80) a. $[\mathbf{R}_i(\mathbf{m}) \wedge \mathbf{G}_i(\mathbf{m})]$

Unlike (78a), (79a) is a simple predication, which means that the truth value is obtained by applying the extension of the predicate **raucht eine Zigarette und sieht fern** [≈ is smoking a cigarette and watching TV] to that of the subject **Maria**. Type-logically speaking, this means that (80a) must be represented by a combination of **m** with the translation of the predicate. The latter would thus have to consist of (80a) minus **m** – or more precisely denote that function which yields (80a) when applied to **m**:[30]

[30]This formulation is misleading in that there are a host of such functions. But then the one that is meant delivers something analogous to (80a) when applied to other constants than **m**.

(81) $|\textbf{raucht und sieht fern}|$
$\equiv$ $(\boldsymbol{\lambda x^e.}[\mathbf{R}_i(\boldsymbol{x}) \wedge \mathbf{G}_i(\boldsymbol{x})])$
$\equiv$ $\underline{(\boldsymbol{\lambda X^{et}.}(\boldsymbol{\lambda Y^{et}.}(\boldsymbol{\lambda x.}[\boldsymbol{X}(\boldsymbol{x}) \wedge \boldsymbol{Y}(\boldsymbol{x})])))}(\mathbf{R}_i)(\mathbf{G}_i)$

Under the assumption that the predicate in (81) results from coordinating **raucht** [≈ smokes] and **sieht fern** [≈ watches TV],the reformulation in the third line brings out how its extension can be determined from the extensions of its three (immediate) parts: by applying the underlined functor to the extensions of the two coordinated predicates. Hence the functor ought to be the type-logical translation of the conjunction **und** [≈ and], which in the interest of clarity we have adorned with a subscript in (82), to distinguish it from the clausal conjunction **und** [≈ and], the interpretation of which is repeated in (83):

(82) $|\textbf{und}_{VP}| = (\boldsymbol{\lambda X^{et}.}(\boldsymbol{\lambda Y^{et}.}(\boldsymbol{\lambda x^e.}[\boldsymbol{X}(\boldsymbol{x}) \wedge \boldsymbol{Y}(\boldsymbol{x})])))$

(83) $|\textbf{und}_{S}| = (\boldsymbol{\lambda v^t.}(\boldsymbol{\lambda u^t.}[\boldsymbol{u} \wedge \boldsymbol{v}]))$ [$\equiv \wedge$; cf. (75a), in Section 5.5]

We skip a the details of an analogous discussion concerning the coordination of two transitive verbs. Both (78b) and (79b) can be represented by (80b), which in turn leads to the decomposition of verb coordination in (84) and ultimately to the interpretation (85) of the corresponding use of **oder** [≈ or]:

(80) b. $[(\exists \boldsymbol{y^e})[\mathbf{Z}_i(\boldsymbol{y}) \wedge \mathbf{K}_i(\mathbf{a}, \boldsymbol{y})] \vee (\exists \boldsymbol{y^e})[\mathbf{Z}_i(\boldsymbol{y}) \wedge \mathbf{M}_i(\mathbf{a}, \boldsymbol{y})]]$
by predicate logic
$\equiv$ $(\exists \boldsymbol{y})[\mathbf{Z}_i(\boldsymbol{y}) \wedge [\mathbf{K}_i(\mathbf{a}, \boldsymbol{y}) \vee \mathbf{M}_i(\mathbf{a}, \boldsymbol{y})]]$

(84) $|\textbf{kauft oder mietet}|$
$\equiv$ $(\boldsymbol{\lambda y^e.}(\boldsymbol{\lambda x^e.}[\mathbf{K}_i(\boldsymbol{x}, \boldsymbol{y}) \vee \mathbf{M}_i(\boldsymbol{x}, \boldsymbol{y})]))$
$\equiv$ $(\boldsymbol{\lambda R^{e(et)}.}(\boldsymbol{\lambda S^{e(et)}.}(\boldsymbol{\lambda y.}(\boldsymbol{\lambda x.}[\boldsymbol{R}(\boldsymbol{x}, \boldsymbol{y}) \vee \boldsymbol{S}(\boldsymbol{x}, \boldsymbol{y})])))) \ (\mathbf{K}_i) \ (\mathbf{M}_i)$

(85) $|\textbf{oder}_{V_{trans}}| = (\boldsymbol{\lambda R^{e(et)}.}(\boldsymbol{\lambda S^{e(et)}.}(\boldsymbol{\lambda y^e.}(\boldsymbol{\lambda x^e.}[\boldsymbol{R}(\boldsymbol{x}, \boldsymbol{y}) \vee \boldsymbol{S}(\boldsymbol{x}, \boldsymbol{y})]))))$

The reformulation given in (80b) is crucial in that it explains the synonymy of (78b) and (79b). A look at the following variants makes this clear:

(78b′) **Alain kauft oder mietet jedes Zimmer.**
[≈ Alain is buying or renting every room.]

(80b′) $(\forall \boldsymbol{y^e})[\mathbf{Z}_i(\boldsymbol{y}) \rightarrow [\mathbf{K}_i(\mathbf{a}, \boldsymbol{y}) \vee \mathbf{M}_i(\mathbf{a}, \boldsymbol{y})]]$
$\not\equiv$ $(\forall \boldsymbol{y^e})[\mathbf{Z}_i(\boldsymbol{y}) \rightarrow \mathbf{K}_i(\mathbf{a}, \boldsymbol{y})] \vee (\forall \boldsymbol{y})[\mathbf{Z}_i(\boldsymbol{y}) \rightarrow \mathbf{M}_i(\mathbf{a}, \boldsymbol{y})]$

(79b′) **Alain kauft jedes Zimmer, oder Alain mietet jedes Zimmer.**
[≈ Alain is buying every room or Alain is renting every room.]

(78b*) **Alain kauft und mietet ein Zimmer.**
[≈ Alain is buying and renting a room.]

(80b*) $(\exists \boldsymbol{y^e})[\mathbf{Z}_i(\boldsymbol{y}) \wedge [\mathbf{K}_i(\mathbf{a}, \boldsymbol{y}) \wedge \mathbf{M}_i(\mathbf{a}, \boldsymbol{y})]]$

$\not\equiv$ $(\exists y^e)[\mathbf{Z}_i(y) \wedge \mathbf{K}_i(\mathbf{a}, y)] \wedge (\forall y)[\mathbf{Z}_i(y) \wedge \mathbf{M}_i(\mathbf{a}, y)]$

(79b*) **Alain kauft ein Zimmer, oder Alain mietet ein Zimmer.**
[≈ Alain is buying a room and Alain is renting a room.]

The paraphrase strategy chosen in (79d) thus fails in these cases. Still the analysis (85) of the use of **oder** [≈ or] for coordinating transitive verbs also works for (78b′):

(86) $|$**Alain kauft oder mietet jedes Zimmer**$|$ by 5.5, (78)

= $|$**kauft oder mietet jedes Zimmer**$|(|$**Alain**$|)$ by 5.5, (89)

= $(\boldsymbol{\lambda} x^e.|$**jedes Zimmer**$|(\boldsymbol{\lambda} y^e.|$**kauft oder mietet**$|(y)(x)))\ (|$**Alain**$|)$ by 5.5, (90)

= $(\boldsymbol{\lambda} x.|$**jedes**$|(|$**Zimmer**$|)(\boldsymbol{\lambda} y.|$**kauft oder mietet**$|(y)(x)))\ (|$**Alain**$|)$ in analogy to clausal coordination in 5.5, (77)

= $(\boldsymbol{\lambda} x.|$**jedes**$|(|$**Zimmer**$|)(\boldsymbol{\lambda} y.|\mathbf{oder}_{V_{trans}}|(|$**kauft**$|)(|$**mietet**$|)(y)(x)))$ $(|$**Alain**$|)$ indirect interpretation of (non-logical) lexical expressions: 5.5, (74)

= $(\boldsymbol{\lambda} x.|$**jedes**$|(\mathbf{Z}_i)(\boldsymbol{\lambda} y.|\mathbf{oder}_{V_{trans}}|(\mathbf{K}_i)(\mathbf{M}_i)(y)(x)))(\mathbf{a})$ indirect interpretation of **jedes** [≈ every]: 5.5, (76d) plus conventions 5.6, (104) and 5.6, (107)

= $(\boldsymbol{\lambda} x.(\boldsymbol{\lambda} Y^{et}.(\boldsymbol{\lambda} X^{et}.(\forall x^e)[Y(x) \rightarrow X(x)]))(\mathbf{Z}_i)(\boldsymbol{\lambda} y.$ $|\mathbf{oder}_{V_{trans}}|(\mathbf{K}_i)(\mathbf{M}_i)(y)(x)))(\mathbf{a})$ by (85)

= $(\underline{\boldsymbol{\lambda} x}.(\boldsymbol{\lambda} Y.(\boldsymbol{\lambda} X.(\forall x)[Y(x) \rightarrow X(x)]))(\mathbf{Z}_i)(\boldsymbol{\lambda} y.$ $(\boldsymbol{\lambda} R^{e(et)}.(\boldsymbol{\lambda} S^{e(et)}.(\boldsymbol{\lambda} y^e.(\boldsymbol{\lambda} x^e.[R(x,y) \vee S(x,y)])))))$ $(\mathbf{K}_i)(\mathbf{M}_i)(y)(x)))(\underline{\mathbf{a}})$ $\boldsymbol{\lambda}$-conversion

≡ $(\underline{\boldsymbol{\lambda} Y}.(\boldsymbol{\lambda} X.(\forall x)[Y(x) \rightarrow X(x)]))(\underline{\mathbf{Z}_i})(\boldsymbol{\lambda} y.$ $(\boldsymbol{\lambda} R.(\boldsymbol{\lambda} S.(\boldsymbol{\lambda} y.(\boldsymbol{\lambda} x.[R(x,y) \vee S(x,y)])))))(\mathbf{K}_i)(\mathbf{M}_i)(y)(\mathbf{a}))$ $\boldsymbol{\lambda}$-conversion

≡ $(\boldsymbol{\lambda} X.(\forall x)[\mathbf{Z}_i(x) \rightarrow X(x)])(\boldsymbol{\lambda} y.$ $(\underline{\boldsymbol{\lambda} R}.(\underline{\underline{\boldsymbol{\lambda} S}}.(\boldsymbol{\lambda} y.(\boldsymbol{\lambda} x.[R(x,y) \vee S(x,y)])))))(\underline{\mathbf{K}_i})(\underline{\underline{\mathbf{M}_i}})(y)(\mathbf{a}))$ two $\boldsymbol{\lambda}$-conversions

≡ $(\underline{\boldsymbol{\lambda} X}.(\forall x)[\mathbf{Z}_i(x) \rightarrow X(x)])$ $(\boldsymbol{\lambda} y.(\boldsymbol{\lambda} y.(\boldsymbol{\lambda} x.[\mathbf{K}_i(x,y) \vee \mathbf{M}_i(x,y)]))(y)(\mathbf{a}))$ $\boldsymbol{\lambda}$-conversion

≡ $(\forall x)[\mathbf{Z}_i(x) \rightarrow (\boldsymbol{\lambda} y.(\underline{\boldsymbol{\lambda} y}.(\boldsymbol{\lambda} x.[\mathbf{K}_i(x,y) \vee \mathbf{M}_i(x,y)]))(\underline{y})(\mathbf{a}))(x)]$ *eigen*-conversion

≡ $(\forall x)[\mathbf{Z}_i(x) \rightarrow (\boldsymbol{\lambda} y.(\underline{\boldsymbol{\lambda} x}.[\mathbf{K}_i(x,y) \vee \mathbf{M}_i(x,y)])(\underline{\mathbf{a}}))(x)]$ $\boldsymbol{\lambda}$-conversion

≡ $(\forall x)[\mathbf{Z}_i(x) \rightarrow (\boldsymbol{\lambda} y.[\mathbf{K}_i(\mathbf{a},y) \vee \mathbf{M}_i(\mathbf{a},y)])(\underline{x})]$ $\boldsymbol{\lambda}$-conversion

≡ $(\forall x)[\mathbf{Z}_i(x) \rightarrow [\mathbf{K}_i(\mathbf{a},x) \vee \mathbf{M}_i(\mathbf{a},x)]]$

In a similar vein, one can show by adapting the analysis (85) of $\mathbf{oder}_{V_{trans}}$ [≈ $\text{or}_{V_{trans}}$] for $\mathbf{und}_{V_{trans}}$ [≈ $\text{and}_{V_{trans}}$] (in an exercise) that the translation of (78b*) is equivalent to the first line of (80b*).

In the cases (78c) and (78d), too, corresponding interpretations of the con-

junction **oder** [≈ or] can be found. A similar reasoning as for the coordination of predicates (78a) and transitive verbs (78b) leads to the following results:

(87) $|\mathbf{oder}_{Det}|$
$= (\boldsymbol{\lambda D^{(et)((et)t)}.(\lambda E^{(et)((et)t)}.(\lambda Y^{et}.(\lambda X^{et}.[D(Y)(X) \vee E(Y)(X)])))})$

(88) $|\mathbf{oder}_{QNP}| = (\boldsymbol{\lambda \mathtt{X}^{(et)t}.(\lambda \mathtt{Y}^{(et)t}.(\lambda X^{et}.[\mathtt{X}(X) \vee \mathtt{Y}(X)]))})$

We skip the derivations of the analyses (87) and (88), and merely test them on the examples:

(89) $|$**Die meisten Deutschen mögen jeden oder keinen Sport**$|$
by 5.5, (88)
$=$ $|$**Die meisten Deutschen**$|(|$**mögen jeden oder keinen Sport**$|)$
by 5.5, (89)
$=$ $|$**Die meisten Deutschen**$|$
$(\boldsymbol{\lambda x^e}.|$**jeden oder keinen Sport**$|(\boldsymbol{\lambda y^e}.|$**mögen**$|(\boldsymbol{y})(\boldsymbol{x})))$
by 5.5, (90)
$=$ $|$**Die meisten**$|(|$**Deutschen**$|)$
$(\boldsymbol{\lambda x}.|$**jeden oder keinen Sport**$|(\boldsymbol{\lambda y}.|$**mögen**$|(\boldsymbol{y})(\boldsymbol{x})))$
by 5.5, (90)
$=$ $|$**Die meisten**$|(|$**Deutschen**$|)$
$(\boldsymbol{\lambda x}.|$**jeden oder**$_{Det}$ **keinen**$|(|$**Sport**$|)(\boldsymbol{\lambda y}.|$**mögen**$|(\boldsymbol{y})(\boldsymbol{x})))$
in analogy to clausal coordination in 5.5, (77)
$=$ $|$**Die meisten**$|(|$**Deutschen**$|)$
$(\boldsymbol{\lambda x}.|\mathbf{oder}_{Det}|(|$**jeden**$|)(|$**keinen**$|)(|$**Sport**$|)(\boldsymbol{\lambda y}.|$**mögen**$|(\boldsymbol{y})(\boldsymbol{x})))$
indirect interpretation of (non-logical) lexical expressions: 5.5, (74)
$=$ $|$**Die meisten**$|(\mathbf{D}_i)(\boldsymbol{\lambda x}.|\mathbf{oder}_{Det}|(|$**jeden**$|)(|$**keinen**$|)(\mathbf{S}_i)$
$(\boldsymbol{\lambda y}.\mathbf{M}_i(\boldsymbol{y})(\boldsymbol{x})))$
by 5.5, (76e), 5.5, (76d) plus conventions 5.6, (104) and 5.6, (107), and 5.5 (76a)
$=$ $\mathbf{MOST}(\mathbf{D}_i)(\boldsymbol{\lambda x}.|\mathbf{oder}_{Det}|(\boldsymbol{\lambda Y^{et}.(\lambda X^{et}.(\forall x^e)[Y(x) \rightarrow X(x)])})$
$(\boldsymbol{\lambda Y^{et}.(\lambda X^{et}.\neg(\exists x^e)[Y(x) \wedge X(x)])})(\mathbf{S}_i)(\boldsymbol{\lambda y}.\mathbf{M}_i(\boldsymbol{y})(\boldsymbol{x})))$ by (87)
$=$ $\mathbf{MOST}(\mathbf{D}_i)(\boldsymbol{\lambda x.(\lambda D^{(et)((et)t)}.(\lambda E^{(et)((et)t)}.}$
$\boldsymbol{(\lambda Y^{et}.(\lambda X^{et}.[D(Y)(X) \vee E(Y)(X)])))}$
$\boldsymbol{(\lambda Y.(\lambda X.(\forall x)[Y(x) \rightarrow X(x)]))}$
$\boldsymbol{(\lambda Y.(\lambda X.\neg(\exists x)[Y(x) \wedge X(x)]))}(\mathbf{S}_i)(\boldsymbol{\lambda y}.\mathbf{M}_i(\boldsymbol{y})(\boldsymbol{x})))$
... several **λ**- and *eigen*-conversions ...
$\equiv$ $\mathbf{MOST}(\mathbf{D}_i)(\boldsymbol{\lambda x}.(\forall y^e)[\mathbf{S}_i(\boldsymbol{y}) \rightarrow \mathbf{M}_i(\boldsymbol{x},\boldsymbol{y})] \vee$
$\neg(\exists y^e)[\mathbf{S}_i(\boldsymbol{y}) \wedge \mathbf{M}_i(\boldsymbol{x},\boldsymbol{y})])$

It is not hard to see that this formula comes close to the paraphrase (79c), for which we still lack the means of interpretation, though – due to the pronoun **sie** [≈ they]. But then one may also check directly that the type-

logical translation of (78c) determined in (89) correctly accounts for the truth conditions of the sentence: roughly speaking, more than 50% of the persons in the extension of **Deutscher** [≈ German] would have to be in the union of the sets of sport-lovers and sport-haters.

The Analysis (88) also leads to the desired result for the sample case (78d):

(90) $|\textbf{Hans besitzt ein Moped oder ein Fahrrad}|$ by 5.5, (78)
= $|\textbf{besitzt ein Moped oder ein Fahrrad}|(|\textbf{Hans}|)$ by 5.5, (89)
= $(\boldsymbol{\lambda x^e}.|\textbf{ein Moped oder}_{QNP}\ \textbf{ein Fahrrad}|(\boldsymbol{\lambda y^e}.|\textbf{besitzt}|(\boldsymbol{y})(\boldsymbol{x})))($
$|\textbf{Hans}|)$ in analogy to clausal coordination in 5.5, (77)
= $(\boldsymbol{\lambda x}.|\textbf{oder}_{QNP}|(|\textbf{ein Moped}|)(|\textbf{ein Fahrrad}|)$
$(\boldsymbol{\lambda y}.|\textbf{besitzt}|(\boldsymbol{y})(\boldsymbol{x})))(|\textbf{Hans}|)$ by 5.5, (90)
= $(\boldsymbol{\lambda x}.|\textbf{oder}_{QNP}|(|\textbf{ein}|(|\textbf{Moped}|))(|\textbf{ein}|(|\textbf{Fahrrad}|))$
$(\boldsymbol{\lambda y}.|\textbf{besitzt}|(\boldsymbol{y})(\boldsymbol{x})))(|\textbf{Hans}|)$ by 5.5, (74)
= $(\boldsymbol{\lambda x}.|\textbf{oder}_{QNP}|(|\textbf{ein}|(\mathbf{M}_i))(|\textbf{ein}|(\mathbf{F}_i))$
$(\boldsymbol{\lambda y}.\mathbf{B}_i(\boldsymbol{y})(\boldsymbol{x})))(\mathbf{h})$ by 5.5, (76b)
= $(\boldsymbol{\lambda x}.|\textbf{oder}_{QNP}|$
$((\boldsymbol{\lambda Y^{et}}.(\boldsymbol{\lambda X^{et}}.(\exists \boldsymbol{x^e})[\boldsymbol{Y}(\boldsymbol{x}) \wedge \boldsymbol{X}(\boldsymbol{x})]))(\mathbf{M}_i))$
$((\boldsymbol{\lambda Y^{et}}.(\boldsymbol{\lambda X^{et}}.(\exists \boldsymbol{x^e})[\boldsymbol{Y}(\boldsymbol{x}) \wedge \boldsymbol{X}(\boldsymbol{x})]))(\mathbf{F}_i))\ (\boldsymbol{\lambda y}.\mathbf{B}_i(\boldsymbol{y})(\boldsymbol{x})))(\mathbf{h})$
by (88)
= $(\boldsymbol{\lambda x}.(\boldsymbol{\lambda \mathtt{X}^{(et)t}}.(\boldsymbol{\lambda \mathtt{Y}^{(et)t}}.(\boldsymbol{\lambda X^{et}}.[\mathtt{X}(\boldsymbol{X}) \vee \mathtt{Y}(\boldsymbol{X})]))))$
$((\boldsymbol{\lambda Y}.(\boldsymbol{\lambda X}.(\exists \boldsymbol{x})[\boldsymbol{Y}(\boldsymbol{x}) \wedge \boldsymbol{X}(\boldsymbol{x})]))(\mathbf{M}_i))$
$((\boldsymbol{\lambda Y}.(\boldsymbol{\lambda X}.(\exists \boldsymbol{x})[\boldsymbol{Y}(\boldsymbol{x}) \wedge \boldsymbol{X}(\boldsymbol{x})]))(\mathbf{F}_i))\ (\boldsymbol{\lambda y}.\mathbf{B}_i(\boldsymbol{y})(\boldsymbol{x})))(\mathbf{h})$
9 $\boldsymbol{\lambda}$-conversions
≡ $(\exists \boldsymbol{x})[\mathbf{M}_i(\boldsymbol{x}) \wedge \mathbf{B}_i(\mathbf{h}, \boldsymbol{x})] \vee (\exists \boldsymbol{x})[\mathbf{F}_i(\boldsymbol{x}) \wedge \mathbf{B}_i(\mathbf{h}, \boldsymbol{x})]$

It should not come as a surprise that the above analyses can be carried out both for the conjunction **and** [≈ and] and for **oder** [≈ or]. We thus get:

(91) $|\textbf{oder}_{VP}| = (\boldsymbol{\lambda Y^{et}}.(\boldsymbol{\lambda X^{et}}.(\boldsymbol{\lambda x^e}.\boldsymbol{X}(\boldsymbol{x}) \vee \boldsymbol{Y}(\boldsymbol{x}))))$ in analogy to (82)

(92) $|\textbf{und}_{V_{trans}}| = (\boldsymbol{\lambda R^{e(et)}}.(\boldsymbol{\lambda S^{e(et)}}.(\boldsymbol{\lambda y^e}.(\boldsymbol{\lambda x^e}.\boldsymbol{R}(\boldsymbol{x}, \boldsymbol{y}) \wedge \boldsymbol{S}(\boldsymbol{x}, \boldsymbol{y})))))$
like (85) & exercise

(93) $|\textbf{und}_{Det}|$ cf. (87)
= $(\boldsymbol{\lambda D^{(et)((et)t)}}.(\boldsymbol{\lambda E^{(et)((et)t)}}.(\boldsymbol{\lambda Y^{et}}.(\boldsymbol{\lambda X^{et}}.[\boldsymbol{D}(\boldsymbol{Y})(\boldsymbol{X}) \wedge \boldsymbol{E}(\boldsymbol{Y})(\boldsymbol{X})]))))$

(94) $|\textbf{und}_{QNP}| = (\boldsymbol{\lambda \mathtt{X}^{(et)t}}.(\boldsymbol{\lambda \mathtt{Y}^{(et)t}}.(\boldsymbol{\lambda X^{et}}.[\mathtt{X}(\boldsymbol{X}) \wedge \mathtt{Y}(\boldsymbol{X})])))$ see (88)

Before turning to the coordination of proper names exemplified in (78e), let us briefly stop and compare the analyses so far. It is conspicuous that they are all of the same form:

- The start with two abstractions, which (hardly surprisingly) correspond to the two arguments, i.e., the extensions of the constituents to be coordinated – schematically:

$$(\boldsymbol{\lambda A}^{c}.(\boldsymbol{\lambda B}^{c}. \ldots$$

It should be noted that the variables abstracted at this point need to be of the same type, for only expressions of the same category can be coordinated.

- Next comes another sequence of abstractions:

  $$(\boldsymbol{\lambda x}^{c_1}.(\boldsymbol{\lambda x}^{c_2}. \ldots$$

  Here the types c_1, c_2, ... match the (potential) arguments of the constituents to be coordinated: in the cases of quantifying NPs (type $\underline{((et)t)}$) and predicates (type $\underline{(et)}$), there is only one argument, with the respective type $c_1 = \boldsymbol{et}$ and $c_1 = \boldsymbol{e}$; in the case of a transitive verbs (type $\underline{e(et)}$), c_1 and c_2 both equal $\boldsymbol{e}$; for determiners (type $\underline{(et)((et)t)}$) we have $c_1 = c_2 = \underline{(et)}$; for ditransitive verbs (type $\underline{e(e(et))}$) one would presumably get $c_1 = c_2 = c_3 = \boldsymbol{e}$; and for attitude verbs (type $\underline{(st)(et)}$), it would have to be $c_1 = \boldsymbol{st}$ and $c_2 = \boldsymbol{e}$. (The last two conjectures will be checked in an exercise!)

- Finally the matrix follows, which is a formula of the form $[\varphi \wedge \psi]$ or, as the (disjunctive) case may be $[\varphi \vee \psi]$, where in φ und ψ the arguments of the coordination get applied to their own potential arguments:

  $$\varphi = \boldsymbol{A}^{c}(\boldsymbol{x}^{c_1})(\boldsymbol{x}^{c_2}) \ldots; \quad \psi = \boldsymbol{B}^{c}(\boldsymbol{x}^{c_1})(\boldsymbol{x}^{c_2}) \ldots.$$

The strategy behind the cross-categorial generalisations of **and** is thus to keep supplying potential arguments to the extensions of the expressions coordinated until they deliver truth values to which ultimately the clausal connective is applied. Thus the two constituents are construed as gappy sentences, as it were, whose gaps are filled immediately: when the coordination is applied to an argument, the latter ends up in both clauses coordinated. Of course, this strategy only works if both constituents are of the same type and if this type is such that it delivers a truth value once all arguments have been supplied; in other words, the type must be of the form $(c_1, (c_2, \ldots t))$. Such types are called *Boolean*[31] or *conjoinable*.

Since the conjoinable types happen to be just those that end in a $\boldsymbol{t}$, the prototypical non-Boolean type is the extension type $\boldsymbol{e}$ of proper names, which thus cannot be captured by the above cross-categorial generalisation. But then how can the coordination in (78e) and its apparent paraphrasability (79e) in terms of Boolean conjunction be explained?

The answer to this question can be found in the type shifting techniques introduced in the previous section. There we showed that, somewhat per-

[31] After the logic 19th century English logic pioneer mentioned in fn. 38 (Section 1.7).

versely, the strategy of obtaining extensions by abstraction can also be applied to proper names – under the pretence that their extensions are unknown. As a result the (prototypical) name **Fritz** then ends up as having a Quantifier extension, i.e., [the characteristic function of] a set of [characteristic functions of] sets of individuals. More specifically we get (for any situation s):

(95) $[\![\mathbf{Fritz}_{Quant}]\!]^s = \lambda X.X(\text{Fritz}); \quad [\![\mathbf{Eike}_{Quant}]\!]^s = \lambda X.X(\text{Eike})$

which translates into indirect interpretation as:

(96) $|\mathbf{Fritz}_{Quant}| = (\boldsymbol{\lambda X^{et}.X(f)}); \quad |\mathbf{Eike}_{Quant}| = (\boldsymbol{\lambda X^{et}.X(e)})$

where **f** and **e** are the ordinary Ty2-translations of **Fritz** and **Eike** as *P*roper *N*ames:

(97) $\mathbf{f} = |\mathbf{Fritz}_{PN}| \in Con_{\boldsymbol{e}}; \quad \mathbf{e} = |\mathbf{Eike}_{PN}| \in Con_{\boldsymbol{e}}$

What is important to notice about the otherwise somewhat awkward and unmotivated analysis (96) is that, unlike the traditional analysis (97), the type it assigns to the extension of the name is the type of quantifying noun phrases – and thus conjoinable. And indeed, on the basis of (96) the coordination in (78e) can be analysed in terms of the above pattern (94) of generalised coordination; and, what is more, it comes out as desired:

(98)
$|\mathbf{Fritz}_{Quant}\ \mathbf{und}_{QNP}\ \mathbf{Eike}_{Quant}\ \mathbf{wohnen\ in\ Tübingen}|$ 5.5, (88)
$= |\mathbf{Fritz}_{Quant}\ \mathbf{und}_{QNP}\ \mathbf{Eike}_{Quant}|(|\mathbf{wohnen\ in\ Tübingen}|)$ in analogy to clausal coordination in 5.5, (77)
$= |\mathbf{und}_{QNP}|(|\mathbf{Fritz}_{Quant}|)(|\ \mathbf{Eike}_{Quant}|)(|\mathbf{wohnen\ in\ Tübingen}|)$ by (94)
$= (\boldsymbol{\lambda}\mathtt{X}^{(et)t}.(\boldsymbol{\lambda}\mathtt{Y}^{(et)t}.(\boldsymbol{\lambda X^{et}}.[\mathtt{X}(\boldsymbol{X})\wedge\mathtt{Y}(\boldsymbol{X})])))(|\mathbf{Fritz}_{Quant}|)(|\ \mathbf{Eike}_{Quant}|)$
$(|\mathbf{wohnen\ in\ Tübingen}|)$ by (96)
$= (\boldsymbol{\lambda}\mathtt{X}.(\boldsymbol{\lambda}\mathtt{Y}.(\boldsymbol{\lambda X}.[\mathtt{X}(\boldsymbol{X}) \wedge \mathtt{Y}(\boldsymbol{X})])))(\boldsymbol{\lambda X^{et}.X(f)})(\boldsymbol{\lambda X^{et}.X(e)})$
$(|\mathbf{wohnen\ in\ Tübingen}|)$ 2 $\boldsymbol{\lambda}$-conversions
$\equiv (\boldsymbol{\lambda X}.[(\boldsymbol{\lambda X.X(f)})(\boldsymbol{X})\wedge(\boldsymbol{\lambda X.X(e)})(\boldsymbol{X})])\ (|\mathbf{wohnen\ in\ Tübingen}|)$ 2 *eigen*-conversions
$\equiv (\boldsymbol{\lambda X}.[\boldsymbol{X(f)} \wedge \boldsymbol{X(e)}])\ (|\mathbf{wohnen\ in\ Tübingen}|)$ $\boldsymbol{\lambda}$-conversion
$\equiv [|\mathbf{wohnen\ in\ Tübingen}|(\mathbf{f}) \wedge |\mathbf{wohnen\ in\ Tübingen}|(\mathbf{e})]$

Readers are invited to convince themselves that the truth conditions of (78e) are indeed captured correctly by this type-logical translation; and while they are at it, they may also work out the details for the disjunctive variant, i.e.:

(78) f. **Fritz oder Eike wohnt in Tübingen.** proper names
[≈ Fritz or Eike lives in Tübingen.]

Moreover, with the unified treatment of proper names and quantifying noun

phrases, it is also possible to account for *mixed coordinations* like the following, which come out adequately:

(78) g. **Fritz oder eine Semantkerin wohnt/-en in Tübingen.**
[≈ Fritz or a female semanticists live/-s in Tübingen.]
proper name + quantifier

h. **Mindesten drei Semantikerinnen und Eike wohnen in Tübingen.**
[≈ At least three female semanticists and Eike live in Tübingen]
quantifier + proper name

The detailed derivation of the truth conditions of (78f-h) is left to the readers as an exercise.

It may be noted in passing that (78e) and (78f) not only differ in the choice of connective, but also in number: while names coordinated with **und** [≈ and] trigger plural agreement, corresponding disjunctions do not (or only marginally so). One may take this contrast between (78e) and (78f) to be a morphosyntactic whim of German (and English alike), if it were not for the fact, that plural agreement in general tends to be concomitant with certain semantic effects that are also observed with names conjoined by **und** [≈ and] – but not if the flank **oder** [≈ or]. In particular, apart from the expected Boolean reading derivable in analogy to (98), the following sentences also have additional (and sometimes more obvious) construals that cannot be so obtained:

(99) a. **Fritz und Eike sind verheiratet.**
[≈ Fritz and Eike are married.]

b. **Fritz und Eike rufen eine Polizeidienststelle an.**
[≈ Fritz and Eike make a phone call to a police station.]

c. **Fritz und Eike wiegen mehr als 100 kg.**
[≈ Fritz and Eike weigh more than 100 kilos.]

According to its Boolean reading, (99a) is already true if both Fritz and Eike are married, no matter who their partners are; the more prominent *reciprocal* reading has it that they are married to each other. This ambiguity can be observed with most verbs whose extensions are always symmetric and which may be used intransitively.[32] – (99b) is most likely understood as saying that Fritz and Eike together call the police. This *collective* reading is not captured by combining the conjunction (94) of $\mathbf{und}_{QNP}$ [≈ and_{QNP}] with the quantifier versions (96) of the proper names **Fritz** and **Eike**. Yet on top of it, (99b) has two distinct Boolean readings, due to scopal interaction of the quantifier **eine Pizza** [≈ a pizza] and the conjunction $\mathbf{und}_{QNP}$ [≈

[32] Adapting standard terminology of mathematical logic, a function R of type ***e(et)*** is called *symmetric* iff $R(x)(y) = R(y)(x)$, for any individuals x and y.

and$_{QNP}$]: on both readings, Fritz and Eike are making separate phone calls, but on only one of them both calls have to be directed to the same police station. The details are sorted out in an exercise. – (99c) may be about Fritz's and Eike's respective weights, or it may be about their joint weight. Only the former reading can be directly captured by (94) and (96), but the other, *cumulative reading* cannot.

Interestingly, the the phenomena observed in (99) can also be observed if the coordinated names are replaced by plural definites like **die Hamms** [≈ the Hamms]; and some persist if a singular, group-denoting description like **das Paar** [≈ the couple] is used instead. We cannot go into these matters here, since a full treatment of the semantic effects of pluralisation lies far beyond the aims of this introductory text. Moreover, as things stand today, these matters are far from settled: precisely how the additional readings in (99) relate to each other and to Boolean conjunction is not fully understood in current semantics – despite numerous attempts of settling the matter and a number of insights gained since the late 1970s.[33] We thus have to leave the matter open and only note that no analogous behaviour can be observed with disjunction for which the cross-categorial account described above seems to be less problematic.[34]

6.5 Adverbial Modification

[to be written]

[33] Primary references include:

- Godehard Link's work on plural; cf. his contribution to the 1991 de Gruyter handbook of *Semantics* (edited by Arnim von Stechow & Dieter Wunderlich).
- Wolfgang Sternefeld's 1998 article on cumulation (though in a slightly different sense from ours) in *Natural Language Semantics* ('Reciprocity and Cumulative Predication').
- Yoad Winter's spectacular reduction of collective readings to Boolean conjunction in his 1996 *Linguistics and Philosophy* article 'A Unified Semantic Treatment of Singular NP Coordination'.

[34] ... which does not mean that the semantic analysis of **oder** [≈ or] is without problems. Interested readers may consult the survey by Maria Aloni in the (online) *Stanford Encyclopedia of Philosophy*.