Coding Predicative Lexemes

Jörg Schuster

12th December 2002

Contents

Chapter 1

Introduction

1.1 About this paper

This is a multi-functional paper. Its original version was a kind of description of and reference for lexicographic work performed for a project called *Biopath*.

Biopath is, roughly, about information extraction from medical and biological texts. One of the problems that one has to face in information extraction is *multiple expressability of states of affairs*. Example: In order to refer to the state of affair that (gene) g activates (protein) p, I can utter phrases of the following form:

- g activates p.
- the activation of p by g
- the p-activator g
- g is a p-activator.
- the p-activating (gene) g
- ...

It would be nice to map all these phrases to some normalized form, e.g. $activate(p, g)$. Yet, in order to be able to do this, one has to have some information about the predicative lexemes *activate*, *activation* and *activator*. In the Biopath project we have coded some information of the required type. In section **??**, I will state explicitly how the coded data helps to improve information extraction from medical and biological texts.

As information about predicative lexemes is useful for other tasks of computational linguistics or linguistics in general, in this paper, I have tried to abstract from the quite specific tasks defined in the Biopath project. So, the current version of this paper tries to give a short overview of the phenomenon of predicativity (in English lexemes) and to be a kind of reference for future coding projects. I.e., if, in the following, I refer to *the project*, I do not mean any specific project. I will use *the project* as an indexical term: It is supposed to refer to that project which the paper is being used for.

This paper has the following structure: Section 1.2 tries to convince you that coding predicative lexemes is an important linguistic task (and that reading the rest of the paper and engaging in predicative-lexeme-coding projects are great activities). It presupposes some familiarity with basic linguistic concepts. If you feel you don't get the point of section 1.2, read chapter 2 first. It is supposed to be a short introduction into semantics and the phenomenon of predicativity. (If you feel that reading chapter 2 is not sufficient for understanding section 1.2, read the rest of the paper and reread section 1.2 or write an email to `js@cis.uni-muenchen.de` and complain.)

Chapter **??** gives a survey of the Sense-Text Model, a linguistic theory founded by Igor Mel'čuk. Chapter **??** gives an introduction into Mel'čuk's Lexical Functions. Knowing the Sense-Text Model is not really necessary for understanding the rest of the paper. But it enhances one's understanding of the Lexical Functions, because these form an integral part of this theory. Lexical Functions are a powerful device for coding properties of predicative lexemes. In the project, we use a modified form of Mel'čukian functions for coding. Chapter **??** defines the way predicative lexemes are coded in the project. Chapter **??** is about applications that use information coded in the project.

1.2 Why code predicative lexemes?

1.2.1 Relevance for semantics in general

Figure 2.1 on page 10 contains several correspondence relations: Among others, there is the relation between linguistic signs (e.g. sentences) and their meaning (relation 1) and the relation between a linguistic sign's meaning (e.g. a formula of predicate logic) and some entity of the real world (e.g. some state of affair). There has been a lot of effort to model this last relation (relation 2). A lot of books have been written about how sentences relate to the world, or, under which circumstances they are true, respectively. Let us call these books *mts-books* (i.e. books about model theoretic semantics), for short. Probably all[1] mts-books disregard modelling relation 1. I.e., they, take [**?**] for an example, use to provide very small (or toy-) lexica and grammars, like the ones given in figures 2.2 and 2.3. With such lexica and grammars it is already possible to model part of relation no. 1 of figure 2.1: One can assign syntactic structures to natural language sentences. Yet, in order to completely model relation 1 of figure 2.1, i.e. to calculate semantic representations, when given natural language sentences as input, one still needs some rules of the kind shown in figure 2.4. Figure 2.5 gives an example of how such a calculation is performed in (I dare to say) one of the best model theoretic approaches, Discourse Structure Theory.

The point of what I am saying about model theoretic semantics is not to critizise it. In any project, one has to disregard *something*. One cannot deal with all problems at one time. Model theoretic semantics uses to disregard relation 1 of figure 2.1 in order to concentrate on relation 2: Its point is to say, for any sentence, under which circumstances it is true and under which it is false.

The point I want to make in this section is that coding predicative lexemes (and all things connected with this activity like defining criteria of predicativity) is an important contribution to modelling relation no. 1 of figure 2.1. There are two major problems whose solution implies coding predicative lexemes: Translation from surface form to some formula (e.g. of predicate logic) and giving an account of synonymy or equivalence. I will argue for this in an instant, but let me make another point first.

Coding predicative lexemes (and activities connected with it) and doing model theoretic semantics should not be viewed as *alternative* projects. I say this explicitly, because a lot of linguists, including Igor Mel'čuk[2], behave as if it was not true. The two projects are often viewed as different linguistic *approaches* and not as different parts of the work semantics has to perform. Yet, on the one hand, relation no. 1 has always to be modelled with regard to relation no. 2: Consider the question if or not the words *all* and *each* should be assigned different meanings. This is a question that arises in modelling relation no.1.Yet, in order to answer this question correctly, one has to model part of relation 2: If *all* can be replaced by *each* in any sentence *s* without changing the truth value of *s*, then *all* and *each* should be mapped to the same meaning. (The same holds, *mutatis mutandis*, for any other pair of words.) On the other hand, modelling relation no. 2 of figure 2.1 always needs input from a model of relation no. 1. For doing mts, one needs to know:

a) which predicates there are (in a given language)

[1] I have never seen an mts-book for which the above sentence is false or exaggerated. But that does not imply that such books do not exist, of course.

[2] He seems to believe that modelling relation no. 2 of figure 2.1 is not a linguistic task, cf. ???

b) the combinatorial properties of the predicates (e.g. what types of arguments they can take and which predicates they can be arguments of)

c) which lexemes or constructions correspond to which predicates or arguments of predicates

To know a) means to have a theory (or at least a kind of criterion) of predicativity. In this paper I will try to do *a little* theory of predicativity. But a) is to difficult and philosophical an issue for this paper. Point b) is about sortal restrictions, like the one that the predicate $love(X, Y)$ cannot have *computer* as its first argument. I will not say much about b). As concerns point c), there are a lot of observations that can be made without knowing much about a) and b): The existence and some properties of the most important predicates can be postulated without presupposing an elaborated theory of predicativity. And it is exactly this postulating work which helps to solve the two major problems, translation from surface to logical form and giving an account of equivalence. Example 1 tries to illustrate this:

Example 1 PREDICATIVE LEXEMES AND LOGICAL FORM OF SENTENCES.
Most people would agree that $resist(X, Y)$ is a two-place predicate. So, we can assign a) the logical form b).[3]

a) Germany fiercely resisted Brazil.

b) $\exists e(resist(g, b, e) \wedge fierce(e))$

Probably some mts-books provide an algorithm for translating a) into b). But, usually, sentences like c) to e) are not considered in mts-books.

c) Germany offered fierce resistance to Brazil.

d) Brazil encountered fierce German resistance.

e) Brazil encountered fierce resistance from Germany.

If we disregard some problems with adjectives and adverbs and the fact that three-place verbs like *offer* do not appear in the lexicon, Discourse Structure Theory as presented in [**?**] is capable of dealing with c) to e): The verb *encounter* is covered by the three dots of rule LI 17 and *Germany* and *Brazil* are covered by LI 11 (cf. 2.2). Yet, DRT as presented in [**?**] would assign d) the form f):

f) $\exists er(encounter(b, r, e) \wedge resistance(r) \wedge german(r) \wedge fierce(r))$

The formula f) is a bad translation for d) in various respects (and the same would be true for c) and e), *mutatis mutandis*). Firstly, *encounter* is treated as the main predicate of the sentence. Yet, *resistance* should be recognized as main predicate; *encounter* is just a support verb. Secondly, the adjective *german* is treated as if it was a property of the resistance. It should be mapped to the first argument of the predicate $resist(X, Y)$. All errors can be summed up by saying that d) (and c) and e)) should be mapped to the same formula as a), namely b), because a), c), d) and e) are all equivalent. So, these sentences illustrate both problems: Translation from surface to logical form and giving an account of equivalence. This can only be done if one codes – in some adequate formal manner – that the predicate $resist(X, Y)$ can be expressed as in a), c), d) and e).

The examples given above are so trivial that a solution can be given in this paper (cf. ???). The great majority of sentences that you can find in real life are a lot more difficult, though. In order to give you an impression of how big the two mentioned problems, translation to logical form and accounting for synonymy, are, I would like to come up with two other examples.

[3] e is an event variable, g and b are constants.

Example 2 TRANSLATION FROM SURFACE TO LOGICAL FORM. The following three sentences are taken from the real world:[4] Try to translate them into predicate logic and you will see that a) you don't have clear intutions about how to do it and b) there is no theory of predicativity strong enough to do the translation with an algorithm:

a) The collection concentrates on the third and last phase of the war, which begins with the military operations initiated by Joan of Arc in 1429.

b) Before that year, the English were on the offensive and a complete and ultimate English victory had become a real threat.

c) With Joan of Arc's campaign the war actually took a turn in favour of the French.

Example 3 ACCOUNTING FOR SYNONYMY. Figure 1.1 is about a normal french sentence: *Le style des persécution policières des gens de lettres en Union Soviétique a évidemment connu, depuis un demi-siècle, des changements sérieux.* ('As every one knows, the way the police persecutes writers in the Soviet Union has changed a lot in the last 50 years.') Guess how many ways there are to paraphrase this sentence. In figure 1.1 this sentence is split into parts A to I and paraphrases are given for each part. If you multiply the numbers of the paraphrases for the parts, you get – more or less – the number of ways to paraphrase the original sentence. 'More or less' because not all paraphrases of parts can be combined with all other paraphrases of parts. On the other hand, not all possible paraphrases are given and purely syntactical paraphrases are not considered.

1.2.2 Relevance for machine translation

It should be clear that the problems discussed in example 1 are also relevant for machine translation. Most systems have great difficulties with sentences whose main predicate is not a verb, but e.g. a support verb construction. I had bablefish[5] translate example 1c) to German. Babelfish returned *Deutschland geleisteter fierce Widerstand nach Brasilien*, which is not even a correct sentence. Such problems can be solved with information as given in ???.

1.2.3 Relevance for information retrieval

As we have seen in example 1, data about predicative lexemes are necessary for calculating the logical form or meaning of phrases. Therefore, such data can also be used for improving various types of information retrieval.

Paraphrasing and string normalization

Figure **??** shows twenty ways to talk about observation events. (There are a lot more, though.) The patterns in the second column describe how the predicate OBSERVE and its arguments are coded at the surface. Having such patterns for each predicate, or at least for the most interesting predicates allows for various types of normalization procedures:

a) As a non-native speaker, one might want to have information about the possible ways of talking about observation events. A paraphrasing routine based on information as described in this paper might look like this:

```
$ prop_paraphrase "John observed an animal"
>>> recognizing:  predicate:  OBSERVE, arg1:  'John', arg2:  'an animal'
>>> Possible sentential prop_paraphrases:
>>> 1.  John made some observations in an animal.
```

[4] In fact, I found them in the internet: http://www.kb.nl/kb/resources/frameset_kb.html?/kb/vak/deelcoll/teksten/jeanne-en.html

[5] http://babelfish.altavista.com/tr

Figure 1.1: A lot of paraphrases of a usual French sentence: *Le style des persécution policières des gens de lettres en Union Soviétique a évidemment connu, depuis un demi-siècle, des changements sérieux.* ('As every one knows, the way the police persecutes writers in the Soviet Union has changed a lot in the last 50 years.'). Multiply the number of paraphrases for each part to get the number of paraphrases for the sentence.

A	B	C	D
bien $\left\{\begin{array}{l}\text{sur}\\ \text{entendu}\end{array}\right\}$			
évidemment			
certainement			
certes			
Il va de soi que	le style de	subir les persécutions $\left\{\begin{array}{l}\text{policières}\\ \text{par la police}\end{array}\right\}$	
Il est $\left\{\begin{array}{l}\text{évident}\\ \text{certain}\\ \text{clair}\\ \text{indéniable}\end{array}\right\}$ que			
chose $\left\{\begin{array}{l}\text{évidente}\\ \text{certaine}\end{array}\right\}$	$\left\{\begin{array}{l}\text{la façon}\\ \text{la manière}\end{array}\right\}$ dont	la police $\left\{\begin{array}{l}\text{persécute}\\ \text{poursuit}\\ \text{harcèle}\end{array}\right\}$	gens de lettres auteurs écrivains
Il n'est pas niable que			
On ne peut nier que			
On est certain que			
15	3	5	3

E	F	G	H	I
	a changé			
	a $\left\{\begin{array}{l}\text{subi}\\ \text{connu}\end{array}\right\}$ des changements			
	des changements sont survenues	bien		
	s'est transformé	beaucoup	au cours de	50 $\left\{\begin{array}{l}\text{années}\\ \text{ans}\end{array}\right\}$
en $\left\{\begin{array}{l}\text{U.R.S.S}\\ \text{Union Soviétique}\end{array}\right\}$	a $\left\{\begin{array}{l}\text{subi}\\ \text{connu}\end{array}\right\}$ des transformations	profondement		
soviétique	des transformations sont survenues	sérieusement	depuis	cinq décennies
	est devenue $\left\{\begin{array}{l}\text{différent}\\ \text{autre}\end{array}\right\}$	de façon $\left\{\begin{array}{l}\text{profonde}\\ \text{sérieuse}\end{array}\right\}$	il y a	demi-siècle
	n'est plus le même	tout à fait	pendant	
	$\left\{\begin{array}{l}\text{a été}\\ \text{s'est}\end{array}\right\}$ modifié	trés [*different*]		
	a $\left\{\begin{array}{l}\text{subi}\\ \text{connu}\end{array}\right\}$ des transformations	tout [*autre*]		
3	15	9	4	4

```
>>> 2.  An animal was observed by John.
>>> 3.  John is an animal observer.
>>> Possible nominal prop_paraphrases:
>>> 1.  John's observation of an animal.
>>> 2.  the observation of an animal by John
>>> 3.  an animal's being observed by John
>>> 4.  the animal observer John
>>> 5.  John, the animal observer
>>> 6.  John, observer of an animal
$ prop_paraphrase "Germany fiercely resisted Brazil"
>>> recognizing:  predicate:  "MAGN(RESIST)", arg1:
>>> 'Germany', arg2:  'Brazil'
>>> Possible sentential prop_paraphrases:
>>> 1.  Germany offered fierce resistance to Brazil.
>>> 2.  Brazil encountered fierce German resistance.
>>> 3.  Brazil encountered fierce resistance from Germany.
>>> Possible nominal prop_paraphrases:
>>> 1.  Germany's fierce resistance against Brazil
>>> ???
```

b) With data of the mentioned type it is also possible to transduce natural language predications to some normalized form. Like so:

Sentence	*Normalized Form*
Leonard was a very close observer of this scene	OBSERVE(1: Leonard, 2: this scene, 3: ?)
he made an observation about the passage on Bernini's St Teresa	OBSERVE(1: he, 2: the passage ... Teresa, 3: ?)
This returns us to Arendt's observation that secrecy is a prerequisite of ...	OBSERVE(1: Arendt, 2: ?, 3: that secrecy ...)

State-of-affair extraction

If natural language predications can be normalized in the way shown above, then the states of affairs in question can of course also be extracted. Suppose that, apart from information about English predicates, we are also in possession of some lists of hyponyms and proper nouns for each English noun. I.e. we are able to match *scientist* to all its hyponyms (such as *linguist*, *logician*, *biologist*, ...) and to a lot of proper nouns of scientists (such as *Igor Mel'čuk*, *Richard Montague*, *Anton Zeilinger*, ...). Then we could write a matching routine of the kind shown in figure 1.2:

Document normalization and classification

Entire documents might be normalized in the way described above. They might then be classified with traditional methods (e.g. the vector space model). I believe that representing documents as sets of states of affairs (of the form OBSERVE(1: bla, 2: bli, 3: blo)) might improve document classification, if the state-of-affair detection is not too bad. The information that word w occurs in text t five times as i-th argument of predicate p and two times as j-th argument of predicate p' is more valuable than the information that w occurs in text t seven times. Layout criteria might be combined with state-of-affair criteria.

Figure 1.2: Using the future matching routine *smatch* (*semantic match*). When smatch will be able to perform tasks as shown below, mainly depends on how many people code how many predicative lexemes with which velocity.

```
$ smatch -corpus=internet -size=10G marry(1: <scientist>, 2:?)

 smatch: Here is a list of persons whom scientists have married
         according to the 10G web pages that I have parsed:

         1. marry(Mr. Bla, Mrs Bli)
            ... the marriage of the famous physician Bla with Mrs Bli ...
            cf. http://bla.bli.org)

         2. marry(1: Mr Blo, Mrs Blu)
            Blo,  married to Blu ...
            (cf. http://blo.blu.com and the file '/Users/bla/scientists.pn')

$ smatch -corpus=gen_corp -size=all activate(1: <enzyme>, 2: bloblase?)

 smatch: Here is a list of enzymes that activate bloblase according to
         the file 'gen_corp':

         1. activate(blibluse, bloblase)
            ... blibluse, an activator of bloblase, was found to bind ...
            cf. 'gen_corp', line 234

         2. activate(blublose, bloblase)
            ... the bloblase activator blublose ...
            cf. 'gen_corp', line 345

         3. activate(blabluse, bloblase)
            ... blabluse binds to and activates bloblase ...
            cf. 'gen_corp', line 456

$ smatch -corpus=internet -size=10G torture(1: <linguist>, 2: <musician>)

 smatch: Sorry, predicate 'torture' not defined.

$
```

Chapter 2

Predicative lexemes

Any talk about predicative lexemes or predicates is a semantic talk. As this paper is not only written for colleagues, but also for non-linguists and beginners, I will be quite explicit and give a very short introduction to semantics.

2.1 Very short introduction to semantics

2.1.1 Basic ideas and concepts

Definition 1 SEMANTICS. Semantics is the theory of the meanings of linguistic expressions.

I.e. semanticists are people who try to write down in a formal and precise way what the expressions of a natural language mean. Suppose John asks you for the meaning of some expression, e.g. *bachelor*. If you answer him, e.g. "A bachelor is an unmarried man", you are already doing semantics.

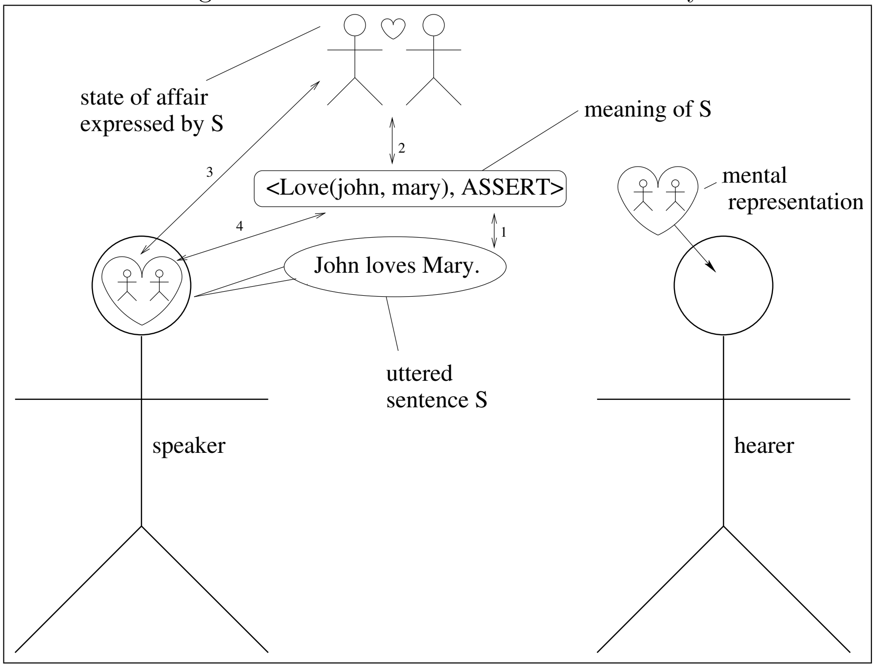
The bachelor example illustrates one first aspect of semantics that might confuse you: The *object* described by semantics is natural language and the *means* with which this object is described is – ultimately – also natural language. I say "ultimately", because professional semanticists do not use to use their *everyday* language for describing linguistic expressions. Normally, they use some artificial *metalanguage* M. In this paper we will also use such an artificial metalanguage. But a metalanguage is only a language if its expressions have a meaning. And the meanings of the metalanguage terms have to be explained by some other terms. I think it is clear that explaining the terms of M in terms of another metalanguage M' does not help, because the terms of M' will have to be explained also. This means, in the end there will always be an explanation of the metalanguage in natural language terms and thus – indirectly – an explanation of natural language terms by natural language terms. As there is no language but language, this kind of circularity is inevitable. But it is not really a vicious kind of circularity: We can avoid it by declaring (the meaning of) some special expressions *basic*, i.e. unexplainable.[1]

In this paper, we will consider *a lot of* expressions basic in meaning. We will, e.g. usually represent the meaning of *John* by *John* and the meaning of *a linguist* by *a linguist*. I.e., we will not make a full-blown semantics, but only consider *some special aspects* of the meaning of natural language expressions. Compare this to a biologist's drawing a picture of the parts of a cell: He might not be interested in the parts of the parts of the cell and therefore represent the nucleus by a circle, the ribosomes by triangles, and so on.

According to definition 1, semantics is the theory of the meanings of linguistic expressions. Why did I say "linguistic expressions" and not just "words"? Well, first of all, there are other

[1] This kind of measure has to be taken in *any* theory: The most basic concepts of a theory T can never be explained by T. If they could, they would not be basic. This phenomenon is a special aspect of what is known as *Münchhausen Trilemma*. It has first been systematically described by the philosopher Hans Albert in [**?**].

Figure 2.1: A talk about John’s love to Mary



linguistics expressions than words, e.g. sentences and texts. Semantics has to deal with them, too. Secondly, semanticists are not really interested in words, but in *lexemes*.

Definition 2 LEXEME. A lexeme is a word form or a series of word forms with a disambiguated (i.e. with one and only one) meaning M, such that M is atomic, i.e. not deducible from the meanings of any of its parts. More formally, a lexeme is a tuple $< F, M >$, where F is the lexeme’s form and M its (atomic) meaning.

Examples 4 LEXEME. *bachelor*$_1$ and *bachelor*$_2$ are two different lexemes, because they differ in meaning: *bachelor*$_1$ = <*bachelor*, “unmarried man”>, *bachelor*$_2$ = < *bachelor*, “person who has the academic degree of a bachelor”>. *lift* and *elevator* are also two different lexemes, because they differ in form: *lift* = <*lift*, “lift”>, *elevator* = <*elevator*, “lift”>. *Prime minister* is one single lexeme because its meaning is not deducible from its parts: Knowing the meaning of *prime* and of *minister* is not sufficient for knowing the meaning of *prime minister*. It is a *multi-word lexeme.*

You may already have thought of the obvious idea that, given the meaning of each lexeme of a language, it should be possible to deduce or calculate the meanings of more complex linguistic expressions, e.g. of sentences. This is indeed one of the goals of semantics. The thesis that this (the deduction of complex meanings from simple meanings) is possible is called *principle of compositionality* or *Frege’s principle*, after the famous mathematician and philosopher *Gottlob Frege* (1848-1925) who first formulated this principle explicitly.

Definition 3 PRINCIPLE OF COMPOSITIONALITY. FREGE’S PRINCIPLE. The meaning of complex expressions E is a function of the atomic expressions A_1, A_2, ... A_i who make up E and of E’s structure.

Probably you wonder why I speak of “the thesis that this is possible”. Isn’t it clear that this is possible? Or, in other words, is Frege’s principle really a thesis? Could it possibly be wrong? The answer is yes, it could be wrong. There are no apriori reasons why languages should not provide atomic signs for each meaning. There might be a natural language in which each meaning is expressed by a single, non-composed sign. In a way, such a language would be quite easy: In

Figure 2.2: A toy lexicon of the type used in a lot of model theoretic papers (from [?, 54-56])

(LI 1)	$\text{DET}_{[\,Num=sing\,]}$	$\rightarrow$	a, every, the, some ...
(LI 2)	$\text{PRO}_{\begin{bmatrix} Num=sing \\ Gen=male \\ Case=+nom \end{bmatrix}}$	$\rightarrow$	he ...
(LI 3)	$\text{PRO}_{\begin{bmatrix} Num=sing \\ Gen=male \\ Case=-nom \end{bmatrix}}$	$\rightarrow$	him ...
(LI 4)	$\text{PRO}_{\begin{bmatrix} Num=sing \\ Gen=fem \\ Case=+nom \end{bmatrix}}$	$\rightarrow$	she ...
(LI 5)	$\text{PRO}_{\begin{bmatrix} Num=sing \\ Gen=fem \\ Case=-nom \end{bmatrix}}$	$\rightarrow$	she ...
(LI 6)	$\text{PRO}_{\begin{bmatrix} Num=sing \\ Gen=-hum \\ Case=-nom/+nom \end{bmatrix}}$	$\rightarrow$	it ...
(LI 7)	$\text{PRO}_{\begin{bmatrix} Num=sing \\ Gen=male/fem/-hum \\ Case=+nom \end{bmatrix}}$	$\rightarrow$	they ...
(LI 8)	$\text{PRO}_{\begin{bmatrix} Num=sing \\ Gen=male/fem/-hum \\ Case=-nom \end{bmatrix}}$	$\rightarrow$	them ...
(LI 9)	$\text{PN}_{\begin{bmatrix} Num=sing \\ Gen=male \end{bmatrix}}$	$\rightarrow$	Jones, Smith, Bill, ...
(LI 10)	$\text{PN}_{\begin{bmatrix} Num=sing \\ Gen=fem \end{bmatrix}}$	$\rightarrow$	Jones, Smith, Mary, [..], Madame Bovary, ...
(LI 11)	$\text{PN}_{\begin{bmatrix} Num=sing \\ Gen=-hum \end{bmatrix}}$	$\rightarrow$	Anna Karenina, Buddenbrooks, [...], ...
(LI 12)	$\text{N}_{\begin{bmatrix} Num=sing \\ Gen=male \end{bmatrix}}$	$\rightarrow$	stockbroker, man, ...
(LI 13)	$\text{N}_{\begin{bmatrix} Num=sing \\ Gen=fem \end{bmatrix}}$	$\rightarrow$	stockbroker, woman, widow, ...
(LI 14)	$\text{N}_{\begin{bmatrix} Num=sing \\ Gen=-hum \end{bmatrix}}$	$\rightarrow$	book, donkey, horse, Porsche, bicycle, ...
(LI 15)	$\text{AUX}_{\begin{bmatrix} Num=sing \\ Fin=+ \end{bmatrix}}$	$\rightarrow$	does
(LI 16)	$\text{AUX}_{\begin{bmatrix} Num=plur \\ Fin=+ \end{bmatrix}}$	$\rightarrow$	do
(LI 17)	$\text{V}_{\begin{bmatrix} Num=sing/plur \\ Trans=+ \\ Fin=- \end{bmatrix}}$	$\rightarrow$	like, love, abhor, own, fascinate, rotate, ...
(LI 18)	$\text{V}_{\begin{bmatrix} Num=sing/plur \\ Trans=- \\ Fin=- \end{bmatrix}}$	$\rightarrow$	stink, rotate, ...
(LI 19)	$\text{V}_{\begin{bmatrix} Num=sing \\ Trans=\beta \\ Fin=+ \end{bmatrix}}$	$\rightarrow$	$\langle Pres,\ sing3^{rd}(\alpha)\rangle$, where $\alpha \in \text{V}_{\begin{bmatrix} Num=sing/plur \\ Trans=\beta \\ Fin=- \end{bmatrix}}$
(LI 20)	$\text{V}_{\begin{bmatrix} Num=plur \\ Trans=\beta \\ Fin=+ \end{bmatrix}}$	$\rightarrow$	$\langle Pres,\ plur(\alpha)\rangle$, where $\alpha \in \text{V}_{\begin{bmatrix} Num=sing/plur \\ Trans=\beta \\ Fin=- \end{bmatrix}}$
(LI 21)	$\text{RPRO}_{\begin{bmatrix} Num=sing/plur \\ Gen=male/fem \end{bmatrix}}$	$\rightarrow$	who ...
(LI 22)	$\text{RPRO}_{\begin{bmatrix} Num=sing/plur \\ Gen=-hum \end{bmatrix}}$	$\rightarrow$	which ...

order to learn it, one would have to simply learn by heart all words of it. There would be no grammar at all. If all languages were non-compositional, machine translation would be a trivial task. Yet, a moment's reflection will convince you that non-compositionality would be a very bad property for a human language: A language dessigned for humans can only have a finite set of expressions, because our mind is finite. If the language is compositional, however, it can express a virtually unbounded, i.e. infinite number of meanings, because the atomic expressions can be combined in infinitely many ways. A language for which Frege's principle does not hold, would only be able to express a finite number of meanings und thus confine our cognitive possibilities crucially. Fortunately, on earth no non-compositional natural language is spoken.

So, for the following deliberations we presuppose that Frege's principle is true. Note, however, that Frege's principle does not imply that finding the true parts of a complex expression is an easy task. In fact, it is a very hard task. In the history of semantics, it turned out that the easiest way to perform this and other tasks, is to start with the examination of *sentences*.

Axiom 1 In semantics, the primary object of examination is the meaning of sentences.[2]

Why should this be a good axiom to work with? There are several reasons: Unlike lexemes, sentences have clear boundaries in written language. We have quite clear intutions where a sentence starts and where it ends. This is so, because sentences are the smallest units which can be used for fully-fledged utterances: Suppose Jack says to you: "love". This would be a fully-fledged lexeme, but not a fully-fledged utterance. You could not know what he wants to tell you. Suppose he says "John's love to Mary". This would be a fully-fledged nominal phrase, but still you could not tell for sure what he means: He could mean that John loves Mary, or he could want you to tell him, if John loves Mary. He could even want to try to change the world with his utterance in such a way that this state of affairs (that John loves Mary) comes into existence.

In all of the cases, Jack should have uttered a sentence. In order to tell you that (according to his opinion) John loves Mary, he should say "John loves Mary". In order to tell you that he wants to know if John loves Mary, he should say "Does John love Mary?". And in order to try to bring about the state of affairs that John loves Mary, he could say something like "John, be in love with Mary!" or "John, you ought to be in love with Mary."

Obviously, there are systematic meaning relations between the three sentences. All three sentences are about the same *state of affairs* (namely John's love to Mary), i.e. they express the same *proposition*, but they bear different *illocutionary force*, i.e. the hearer is supposed to react in different ways. In the first case (assertion) he is supposed to assume that the state of affairs is a fact. In the second case, he is supposed to tell the speaker whether the state of affairs is a fact or not. In the third case, he is supposed to *make* the state of affairs a fact.

Definition 4 SPEECH ACT (TYPE). ILLOCUTIONARY TYPE. A type of act which is performed by uttering linguistic expressions (mostly sentences). Examples for speech acts: Assertion, question, command.

Definition 5 ILLOCUTIONARY FORCE. That part of the meaning of a sentence S which determines the speech act that can be performed with S.

Definition 6 PROPOSITION. That part of the meaning of a sentence which determines what state of affairs the sentence is about. I.e. propositions are linguistic *representations* or *pictures* of states of affairs. That is why their most interesting property is their *truth value* (i.e. whether they are true or false). Strictly speaking, propositions are the only entities that can be said to be true or false. (Assertions or statements are said to be true if the proposition they express is true.)

Definition 7 STATE OF AFFAIRS. A possible fact. Something which can be said to be the case. A state of affairs that is the case ("that exists") is called a *fact*.

From these definitions it ought to be clear how the meanings of sentences look like: They consist of a proposition and an illocutionary force.

Definition 8 MEANING OF SENTENCES. A sentence's meaning is a tuple $< P, I >$, where P is a (possibly complex) proposition (like $love(john, mary)$) and I is an illocutionary force (like ASSERT, ASK or COMMAND).

In this section we have introduced some of the most basic ideas and concepts of semantics. Let us now have a closer look at some aspects of them with the help of a sentence that is very frequent in linguistic papers: *John loves Mary.*

[2] This statement is of another category than Frege's principle. It is not a thesis that might be true or false, but rather an almost never explicitly stated guiding principle which almost all semanticists follow. Usually, *practical reasons* are decisive for a linguist's adopting such a guiding principle. I state some of these guiding principles explicitly and I call them *axioms*, though they do not have much in common with mathematical axioms.

Figure 2.3: A toy grammar of the type used in a lot of model theoretic papers (from [**?**, 53-54])

(PS 1)	$S_{\begin{bmatrix} Num=\alpha \end{bmatrix}}$	$\rightarrow$	$NP_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \\ Case=+nom \end{bmatrix}}$	$VP'_{\begin{bmatrix} Num=\alpha \\ Fin=+ \end{bmatrix}}$
(PS 2)	$S_{\begin{bmatrix} Num=\alpha \\ Gap=NP_{Num=\gamma} \end{bmatrix}}$	$\rightarrow$	$NP_{\begin{bmatrix} Num=\{\alpha,\gamma\} \\ Gen=\beta \\ Case=+nom \\ Gap=NP_{Num=\gamma} \end{bmatrix}}$	$VP'_{\begin{bmatrix} Num=\alpha \\ Fin=+ \\ Gap=- \end{bmatrix}}$
(PS 3)	$S_{\begin{bmatrix} Num=\alpha \\ Gap=NP_{Num=\gamma} \end{bmatrix}}$	$\rightarrow$	$NP_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \\ Case=+nom \\ Gap=- \end{bmatrix}}$	$VP'_{\begin{bmatrix} Num=\alpha \\ Fin=+ \\ Gap=NP_{Num=\gamma} \end{bmatrix}}$
(PS 4)	$VP'_{\begin{bmatrix} Num=\alpha \\ Fin=+ \\ Gap=\gamma \end{bmatrix}}$	$\rightarrow$	$AUX_{\begin{bmatrix} Num=\alpha \\ Fin=+ \end{bmatrix}}$	**not** $VP_{\begin{bmatrix} Num=\delta \\ Fin=- \\ Gap=\gamma \end{bmatrix}}$
(PS 5)	$VP'_{\begin{bmatrix} Num=\alpha \\ Fin=+ \\ Gap=\gamma \end{bmatrix}}$	$\rightarrow$	$VP_{\begin{bmatrix} Num=\alpha \\ Fin=+ \\ Gap=\gamma \end{bmatrix}}$	
(PS 6)	$VP'_{\begin{bmatrix} Num=\alpha \\ Fin=\beta \\ Gap=\gamma \end{bmatrix}}$	$\rightarrow$	$V_{\begin{bmatrix} Num=\alpha \\ Fin=\beta \\ Trans=+ \end{bmatrix}}$	$NP_{\begin{bmatrix} Num=\gamma \\ Gen=\delta \\ Case=-nom \\ Gap=\gamma \end{bmatrix}}$
(PS 7)	$VP_{\begin{bmatrix} Num=\alpha \\ Fin=\beta \end{bmatrix}}$	$\rightarrow$	$V_{\begin{bmatrix} Num=\alpha \\ Fin=\beta \\ Trans=- \end{bmatrix}}$	
(PS 8)	$NP_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \\ Case=\gamma \\ Gap=NP_{Num=\alpha} \end{bmatrix}}$	$\rightarrow$	ϕ	
(PS 9)	$NP_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \\ Case=\gamma \end{bmatrix}}$	$\rightarrow$	$DET_{\begin{bmatrix} Num=\alpha \end{bmatrix}}$	$N_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \end{bmatrix}}$
(PS 10)	$NP_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \end{bmatrix}}$	$\rightarrow$	$PN_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \end{bmatrix}}$	
(PS 11)	$NP_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \\ Case=\gamma \end{bmatrix}}$	$\rightarrow$	$PRO_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \\ Case=\gamma \end{bmatrix}}$	
(PS 12)	$NP_{\begin{bmatrix} Num=plur \\ Gen=\beta \\ Case=\gamma \end{bmatrix}}$	$\rightarrow$	$NP_{\begin{bmatrix} Num=\delta \\ Gen=\epsilon \\ Case=\gamma \end{bmatrix}}$ **and**	$NP_{\begin{bmatrix} Num=\eta \\ Gen=\theta \\ Case=\gamma \end{bmatrix}}$
(PS 13)	$N_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \end{bmatrix}}$	$\rightarrow$	$N_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \end{bmatrix}}$	$RC_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \end{bmatrix}}$
(PS 14)	$RC_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \end{bmatrix}}$	$\rightarrow$	$RPRO_{\begin{bmatrix} Num=\alpha \\ Gen=\beta \end{bmatrix}}$	$S_{\begin{bmatrix} Num=\gamma \\ Gap=NP_{Num=\alpha} \end{bmatrix}}$

2.1.2 A talk about John's love to Mary

Figure 2.1 represents the entities involved in a talk about John's love to Mary. Any talk involves some entities that are directly observable and some other entities whose existence has to be postulated. The observable entities include the speaker, the hearer and an uttered sentence S, in our example the sentence *John loves Mary.* The entities that are not directly observable include the meaning of S, the state of affairs which S describes and some representations of S's meaning (or of the state of affairs) in the minds of hearer and speaker. As described in definition 8, the meaning of S consists of a proposition ($love(john, mary)$, in our example) and an illocutionary force (ASSERT, in our example). The proposition of our sentence is the same as the proposition of the sentences *Does John love Mary?* and *John, be in love with Mary!*.

Note that the talk also includes some correspondence relations (represented by arrows): The first and, from a linguistic point of view, most important correspondence is the one between S and its meaning. For each element of S, there is a corresponding element in the (propositional part of the) meaning of S: The verb form *loves* corresponds to the predicate $love(,)$. (The term *predicate* will be discussed and defined below.) The proper names *John* and *Mary* correspond to the predicate's *arguments* (to be defined below) *john* and *mary*. Correspondence relation no. 2 is the one between S's meaning and the state of affairs. For each element of S's proposition, there is an element of the state of affairs. The predicate $love(,)$ corresponds to the heart (in real life: to the love), the arguments *john* and *mary* correspond to the stick-figures (in real life: to the persons John and Mary). For the illocutionary force there is no corresponding element. Correspondence relations no. 3 and 4 are the ones between state of affair and mental representation and between mental representation and meaning. From a liniguistic point of view they are least important. Apart from that, they are the least understood of all correspondence relations. I will say a few words about them first.

The correspondence relations between mental representation, state of affair and meaning

Science has already looked into people's brains, but it has not observed anything like hearts or stick-figures in it. It has observed, though, other kinds of entities like neurons and their behaviour, e.g. neuronal firing patterns. This paper (and linguistics in genereral) is not about our mental representations of real or fictive entities. We do not care how they look like. We are just forced to believe that some type of mental representations or other has to exist. For, what is the point of an assertion? Its point is to *inform* the hearer about some (alleged) fact. This means: The hearer is supposed to lack some piece of information/mental representation (in our case, the information that John loves Mary) *before the assertion.* And he is supposed to have the piece of information in question in his set of information units *after the assertion.* In short, the speaker is supposed to make the hearer come to know *more* about the world. In our naive picture, this *making someone come to know more about the world* is done by giving him a (new) mental representation. We picture assertion, or linguistic communication in general, as the process of inducing (new) mental representations in the hearer. I say "inducing", because we cannot simply *give* people pieces of our mind. We have to give them some linguistic expression, usually a sentence. They have to "process" the expression in order to understand its meaning and to form the same, or, at least, a similar piece of mind, if they want to.

That is why we postulate correspondence relations between S's meaning, the mental representations of speaker and hearer and the state of affairs: Speaker and hearer have a concept of love (represented by the heart in the speaker's head and by the heart approaching the hearer's head) and they have concepts of John and Mary. They can "assemble" these concepts in a way similar to the way meanings like $love(,)$, *john* and *mary* can be assembled. I.e. mental representations can be *translated* into meanings (which can be translated into surface form expressions) which correspond to states of affairs.

You might object that it is not really necessary to postulate the existence of states of affairs: The point of talking is talking about *facts.* So why are facts not part of the picture instead of states of affairs? And: What is the difference between states of affairs and mental representations? Or: What is the difference between states of affairs and propositions? Do they not coincide?

First, linguistic communications are neither *about* mental representations nor about propositions, they are about (alleged) facts. If I tell you that the unicorn has been eaten by a dragon, then I do neither want to tell you something about my mental representations, nor about the meaning (e.g. the proposition) of my sentence. Rather, by uttering my sentence, I *give* you a proposition *with the objective* that you form a new mental representation *about* a unicorn and a dragon. Yet, as there are neither unicorns nor dragons, I cannot possibly communicate a fact with my statement. However, I do make a well formed English utterance, if I say "The dragon ate the unicorn." So, I do refer to *something*. We could call this something "fact according to the fairy tale X" or "alleged fact". But these terms would be confusing, because they imply that the something in question is not the case/and or is true according to a certain fairy tale/speaker. And we don't want to imply that either. We need a term that covers all possible cases: The something is a fact, the something is not a fact, we don't know, if the something is a fact, but it is a fact according to X, and so on. Therefore, we use the term *state of affairs*.

Note that figure 2.1 involves three levels of discussion: A psychological level, an ontological level and a semantic level.

On the psychological level, questions of the following kind might be discussed:

- Which concepts do speakers of a certain language L have?
- Are there language independent concepts?
- Are all concepts language independent?
- How are concepts physically/neurologically realized ("implemented")?
- ...

Definition 9 ONTOLOGY. METAPHYSICS. [3] Theory about what types of entities there are (in a certain domain of interest).[4]

On the ontological level, questions of the following kind might be discussed:

- What is the relation between words and things: Which words do not point to anything in the outside world (consider e.g. *dog*, *love*, *number*, *unicorn*, *alien*, *God*)?
- Might it be the case that language does not point to anything, i.e. that language is not an adequate means to describe the world as it really is?
- ...

Note that ontology is also concerned with the *truth* of sentences. If a sentence S is true depends on the existence of certain entities and/or on whether certain states of affairs are facts or not. Example: If the sentence *John loves Mary* is true depends on a) whether or not John, Mary and the love relation exist and b) whether or not John loves Mary (i.e. whether or not the state of affairs that John loves Mary is a fact).

Semantics is not about questions of type a). But it usually tries to deal with questions of type b). Semantics tries to model relation no. 2 of figure 2.1 without thereby doing too much ontology.

Problem 1 The fact that semantics tries to model relation no. 2 of figure 2.1 on page 10 raises two questions:

a) Why should semantics want to do that?

[3]Some people do not consider *ontology* and *metaphysics* synonyms.

[4]Note that figure 2.1 is an ontology: It is a theory about what entities are involved in a communication event.

b) How can it be done (without getting involved with too much ontology)?

Concerning problem 1a), I can only give some hints here.[5] The answer has to do with Axiom 2.

Axiom 2 "Einen Satz verstehen heißt, wissen was der Fall ist, wenn er wahr ist." [**?**, 4.024] (Translation (js): *To understand a sentence means to know what is the case, when it is true.*)

As *understand a sentence* means almost the same as *know the meaning of a sentence*, we get the following axiom:

Axiom 3 A good way to find out (part of) the meaning of a sentence is to ask under which circumstances it is true.

I would like to give one example to illustrate these axioms:

Consider the meaning of the conjunctions *and* and *or*? How would you characterize their meaning? The first thing you could say is that they can be used to connect sentences. But this does not really help to tell apart the meanings of *and* and *or*. *Both* can be used to connect sentences, yet they are clearly different in meaning. A moment's reflection will convince you that their difference in meaning has to do with the truth values of the connected sentences: A sentence of the form A *and* B is true if and only if (*iff*, for short) A is true and B is true. A sentence of the form A *or* B is true iff either A or B or both are true. So, we can describe the meaning of these words in terms of relation no. 2 of figure 2.1. In other words, only if we speek of relation 2, i.e. of the truth of sentences, can we adequately describe the meaning of *and* and *or*.

We cannot go into more details. Suffice it to say that for describing certain aspects of a sentence's meaning it is necessary to describe the circumstances under which the sentence is true.

Concerning problem 1b): In order to not get involved with too much ontology (or, with *all of the world*, respectively), (model theoretic) semantics uses a quite simple trick: It makes small descriptions of part of the world called *models*. These models are not claimed to be pictures of the real world. All semantics claims about any model is that *if* it was a picture of the real world, then certain sentences would be true and certain other sentences false. For more details confer figure 2.6.

Modelling the relation no. 2 of figure 2.1 is one important task of semantics. A lot of books have been written about how to do it (cf. e.g. [**?**]). Much less books have been written about how to model relation 1, although this task has to be done first. I dare to say that nowadays the lack of information about relation 1 is one of the main handicaps for a lot of linguistic theories and applications. The rest of this paper is dedicated to modelling relation 1.

The correspondence relation between surface form and meaning

In the last section we discussed a toy ontology or model. The most important implication of our model (if interpreted as an ontology) is that there are basically two types of things in the world: relations and individuals. Relations can be compared to games and individuals to players. Like games, relations are *unsaturated* entities. They need participants. A game of chess is only a game of chess if two players play it. A soccer match is only a soccer match if 22 players, each with a certain part, play it. Similarly, a love is only a love if there is a lover and a beloved one. A murder is only a murder if there is a murderer and a victim. An intelligence is always someone's intelligence. A sale is always a sale of something by someone to someone for some price. Note that individuals do not have to be concrete entities (like humans and cars). They may also be abstract (like numbers and laws). Of course, abstract individuals cannot act all the parts that concrete individuals can act. (John may love Mary and the number 2, but the number 2 cannot love him.)

Semantics usually presupposes that there is an isomorphy, a correspondence between ontological and semantic categories: I have already claimed that propositions (semantic category)

[5] An interesting discussion of this and similar topics is given in [**?**, 7-24].

are pictures of states of affairs (ontological category). There are also special pictures for relations and individuals[6], namely *predicates* and *semantic names*[7]. The idea is that predicates are, just like relations, unsaturated entities which assign certain roles. Semantic names, the semantic counterparts of individuals, can play these roles.

This idea goes back to Frege: He construed (the meaning of) sentences as consisting of a function (i.e. the predicate of the sentence) and its argument (the subject):

> Behauptungssätze im allgemeinen kann man ebenso wie Gleichungen oder analytische Ausdrücke zerlegt denken in zwei Teile, von denen der eine in sich abgeschlossen, der andere ergänzungsbedürftig, ungesättigt ist. So kann man z.B. den Satz "Caesar eroberte Gallien" zerlegen in "Caesar" und "eroberte Gallien". Der zweite Teil ist ungesättigt, führt eine leere Stelle mit sich, und erst dadurch, dass diese Stelle von einem Eigennamen ausgefüllt wird oder von einem Ausdrucke, der einen Eigennamen vertritt, kommt ein abgeschlossener Sinn zum Vorschein. Ich nenne auch hier die Bedeutung dieses ungesättigten Teiles Funktion. In diesem Falle ist das Argument Caesar. [**?**, 29]

In this passage, Frege still analyses (the meaning of) verb *and* direct object as predicate and considers the subject the only argument of the predicate. Nowadays, we would analyse *erobern* ("conquer") as a two-place predicate and ascribe the following logical structure to Frege's sample sentence:

$$Conquer(caesar, gaul)$$

Note that this logical formula is supposed to be a representation of the *meaning* of *Caesar conquers Gaul*. I.e. we represent the meaning of the verb *conquer* by

$$Conquer(\_,\_)$$

and the meanings of *Caesar* and *Gaul* by the words *caesar* and *gaul*. The free places in the meaning representation of *conquer* are called *argument places*. In order to make a complete term out of a predicate, we have to fill its argument places.

Definition 10 PREDICATE. An n-place predicate is a meaning with n empty places, called *argument places*, that have to be filled with other meanings.

Definition 11 ARGUMENT. OPERATOR. Let L be a predicative lexeme. If (the meaning of) lexeme A fills one of L's argument places, then (the meaning of) A is called an argument of L. For any pair of lexemes L_1 and L_2: If L_1 is an argument of L_2, then L_2 is called operator of L_1. I.e. *argument* and *operator* are simply converse concepts. Example: In *John loves Mary*, *John* and *Mary* (or the meanings of these expressions) are arguments of *love* (or LOVE) and *love* (or LOVE)is an operator of *John* and of *Mary* (or of their meanings, respectively)

Definition 12 PREDICATIVE LEXEME. A lexeme whose meaning is a predicate.

Notational convention 1 ARGUMENT PLACE. In the following, we will not represent argument places by underscores, but either by arabic numerals or by arabic numerals followed by a description of the role taken by the lexemes which fill the argument place. The predicate itself always recieves the argument number 0. (Therefore, we will not always write it.) Examples: see notational convention 2.

Notational convention 2 PREDICATE. From now on we will write predicates in small capital letters either in bracket notation, like so:

[6] Note that this metaphysics is quite naive. The world doesn't have to be isomorphic to language. Yet, in the history of semantics, it turned out that assuming such a naive metaphysics is very useful.

[7] This term has been introduced by Mel'čuk in [**?**, 4]

0: LOVE(1: [lover], 2: [beloved one]))

or in tree notation, like so:

$$\begin{tikzcd} & \text{LOVE} \arrow[dl, "1"'] \arrow[dr, "2"] & \\ \text{[lover]} & & \text{[beloved one]} \end{tikzcd}$$

Notational convention 3 MEANING. We write $\|L\|$ to denote the propositional part of L's meaning (cf. figure 2.1).

Examples 5 PREDICATE. Almost all verbs have a predicate as their meaning. (We will discuss those verbs which do not express a predicate in section **??**.)

a) $\|sell\|$ = SELL(1: [seller], 2: [good], 3: [buyer], 4: [price])
{John}1 {sold}0 {a car}2 {to Mary}3 {for €2000}4.

b) $\|observe\|$ = OBSERVE(1: [observer], 2: [observed entity], 3: [observed state of affairs])
{John}1 {observed}0 {in animals}2 {that they didn't like the food}3.

c) $\|believe\|$ = BELIEVE(1: [believer], 2: [believed state of affairs])
{John}1 {believes}0 {that Jack loves Mary}2

d) $\|sleep_v\|$ = SLEEP(1: [sleeper])
{John}1 {sleeps}0

For many verbs there is a noun with the same propositional meaning. Such nouns are called *predicative nouns*.

e) $\|sale\| = \|sell\|$ = SELL(1: [seller], 2: [good], 3: [buyer], 4: [price])
{John's}1 {sale}0 {of a car}2 {to Mary}3 {for €2000}4

f) $\|observation\| = \|observe\|$ = OBSERVE(1: [observer], 2: [observed entity], 3: [observed state of affairs])
{John's}1 {observation}0 {in animals}2 {that they didn't like the food}3.

g) $\|belief\| = \|believe\|$ = BELIEVE(1: [believer], 2: [believed state of affairs])
{John's}1 {belief}0 {that Jack loves Mary}2

h) $\|sleep_n\| = \|sleep_v\|$ = SLEEP(1: [sleeper])
{John's}1 {sleep}0

Almost all adjectives have a 1-place predicate as meaning:

i) $\|green\|$ = GREEN(1: [green entity])
{The table}1 {is green}0

j) $\|beautiful\|$ = BEAUTIFUL(1: [beautiful entity])
{John}1 {is beautiful}0

k) $\|intelligent\|$ = INTELLIGENT(1: [intelligent entity])
{Mary}1 {is intelligent}0

There are also some adjectives which express an n-place ($n > 1$) predicate:

l) $\|jealous\|$ = JEALOUS(1: [jealous entity], 2: [object of 1's jealousy])
{John}1 {is jealous}0 { of Mary}2

Figure 2.4: An algorithm and one "construction rule" for modelling relation no. 2 of figure 2.1, i.e. for calculating a semantic representation from a syntactic representation (from [**?**, 86,121])

DRS-Construction Algorithm

Input: a discourse D = $S_1, \ldots, S_i, S_{i+1}, \ldots, S_n$ the empty DRS K_0

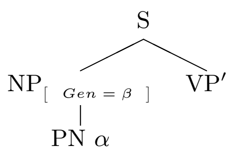
Keep repeating for $i = 1, \ldots, n$**:**

(i) add the syntactic analysis $[S_i]$ of (the next) sentence S_i to the conditions of K_{i-1}; call this DRS K_i^*. Go to (ii).

(ii) Input: a set of reducible conditions of K_i^*. Keep on applying construction principles to each reducible condition of K_i^* until a DRS K_i is obtained that only contains irreducible conditions. Go to (i).

CR.PN

Triggering configuration $\gamma \subseteq \bar{\gamma} \in$ **Con**$_k$



m) $\|proud\|$ = PROUD(1: [proud entity], 2: [object of 1's pride])
{John}1 {is proud}0 { of Mary}2

Prepositions can be predicative lexemes, too. Yet, their analysis involves some complications. Therefore, we discuss them in a section of its own (**??**).

Definition 13 SEMANTIC NAME. In this paper[8], we will call semantic names (the meanings of) those expressions that can only act as arguments, but never as predicates. (Terminology according to Mel'čuk, cf. [**?**, 4].) Note: If the meaning of an expression e is an X, then e itself is not therefore an X, too. But, in order to not introduce too many terms, I will use *semantic name* for meanings $\|L\|$ and for corresponding expression L.

Remark 1 SEMANTIC NAME. In this paper, *common nouns* like *linguist*, *unicorn*, and so on, semantic names, although they are analyzed as predicates in logic. The reasons for this are a little complicated. I will discuss them in section **??**.

These definitions allow for a new definition of the terms *proposition* and *state of affair* with the predicate-argument and with the relation-individual dichotomy.

[8] With definitions that start with *In this paper* I introduce terms, concepts or ways of talking that are not really common in linguistics. I don't claim they *ought* to be common. It is for practical and didactic reasons that I introduce them. In a lot of cases, there are plenty of different terms which mean more or less the same as the term I use. By introducing new or uncommon terms I avoid tedious terminological discussions.

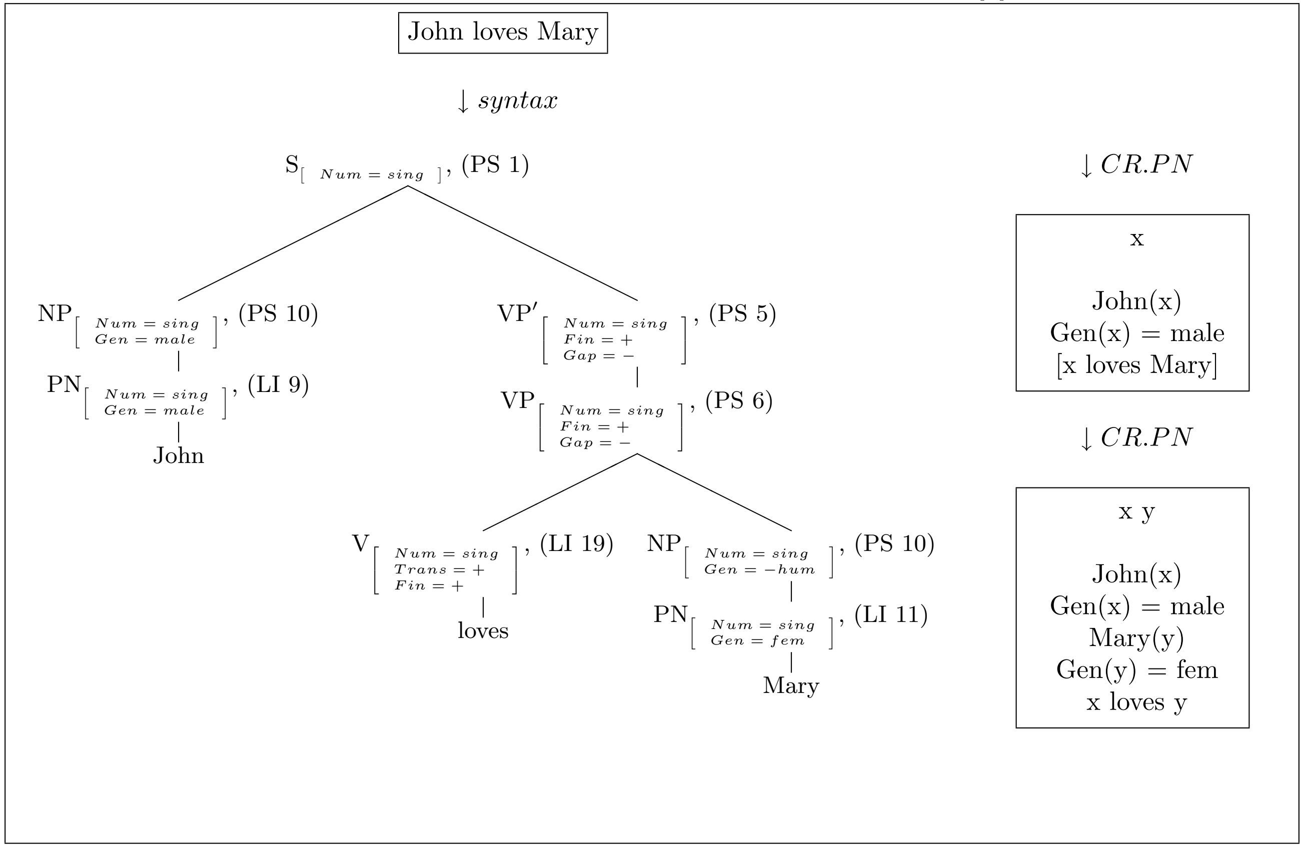
Definition 14 RECURSIVELY FILLED. The argument places of an unsaturated entity are recursively filled iff its argument places are filled and if the argument places of each of its arguments are recursively filled. Cf. examples 6

Definition 15 PROPOSITION. In this paper, any predicate whose argument places are recursively filled is called a proposition.

Definition 16 STATE OF AFFAIR. In this paper, any relation (denoted by a predicate) whose argument places are recursively filled is called a state of affair.

As stated in figure 2.1 and in definition 8, the meaning of sentences includes a possibly complex proposition and an illocutionary force. From definitions 15 and 14 it should be clear that having predicates as arguments is one way for a proposition to be complex. The following examples illustrate this type of complexity of propositions. (We will consider other types of complexity of propositions in section 2.3.2). Besides, the examples illustrate a phenomenon that will play an important role in the following sections: Not only sentences, but also nominal phrases express propositions. Example: The sentence *John loves Mary* expresses the same proposition as the nominal phrase *John's love to Mary*. I.e., we can say that the verb $love_v$ has *almost* the same meaning as the noun $love_n$. The only difference is that one generates *a sentence* by filling $love_v$'s argument places, whereas the filling of $love_n$'s argument places results in a nominal phrase. And only sentences can bear an illocutionary force. With an NP (nominal phrase) I can code states of affairs, but I cannot tell hearers what they are supposed to do with the states of affairs (believe them, say whether they are facts, make them facts ...). I.e. sentences and NPs can never be *fully* synonymous. They can only be synonymous with respect to the expressed proposition. We call this type of synonymy *p-synonymy*.

Figure 2.5: Modelling relation 1 of figure 2.1 according to [?]

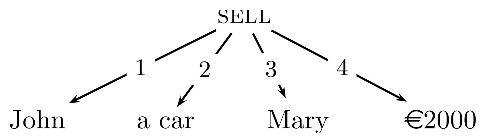


Definition 17 P-SYNONYMOUS. In this paper, two expressions e_1 and e_2 are called p-synonymous iff they express the same proposition or the same predicate.

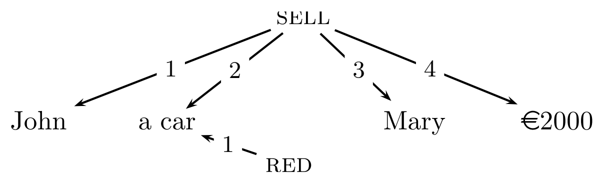
Notational convention 4 P-SYNONYMOUS. If e_1 and e_2 are p-synonymous, we write $e_1 \sim e_2$. Examples: *Mary is in Munich* $\sim$ *Mary's being in Munich. activation* $\sim$ *activate.*

Examples 6 PROPOSITION.

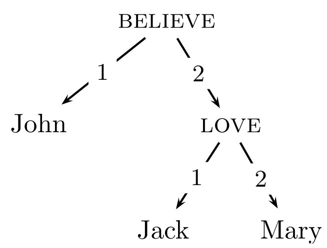
a) John sells a car to Mary for € 2000. $\sim$
John's sale of a car to Mary for € 2000



b) John sells a red car to Mary for € 2000. $\sim$
John sells a car which is red to Mary for € 2000. $\sim$
John's sale of a red car to Mary for € 2000 $\sim$
John's sale of a red car which is red to Mary for € 2000

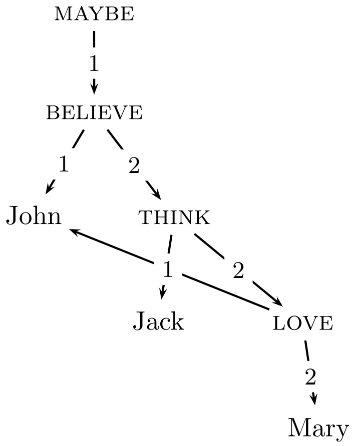


c) John believes that Jack loves Mary. $\sim$
John believes in[9] Jack's love to Mary. $\sim$
John's belief that Jack loves Mary $\sim$
John's belief in Jack's love to Mary



d) Maybe, John_1 believes that Jack thinks that he_1 loves Mary. $\sim$
It is possible that John_1 believes that Jack thinks that he_1 loves Mary. $\sim$
John_1 might believe that Jack thinks that he_1 loves Mary. $\sim$
... $\sim$
John_1's possible belief that Jack thinks that he_1 loves Mary

[9]There is of course a reading of *believe in* whose meaning differs from *believe that*, as in *I don't believe in Generative Syntax*. This reading is not to be considered here.



Examples 6 illustrate some points that still have to be discussed:

a) If you are familiar with predicate logic, you will have noticed that the trees in examples 6 are *almost* equivalent to logical formulas. *Almost*, because *quantifiers* are left untranslated. Quantifiers are words like *a*, *the*, *all*, *each*. In logic they are represented by certain operators. Example 6a would be translated to a formula like the following:

$$\exists x(car(x) \wedge sell(John, x, Mary, €2000)$$

In our meaning representations, we simply disregard quantifiers, because they are not relevant for the objectives of this paper. They could, however, be introduced, if necessary.

b) Example 6b illustrates one type of propositional complexity. Here two predicates share one argument, but neither of the predicates is an argument of the other. It also illustrates that the number of ways to express a proposition usually rises with its complexity.

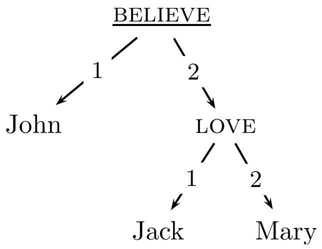
c) Example 6c illustrates another type of propositional complexity: Predicates may take other predicates as their arguments. Note, however, that example 6c is a valid proposition in the sense of definition 15, because its argument places are recursively filled.

d) Example 6d contains the predicate MAYBE. It differs from the predicates discussed so far in that all its argument places (in this special case: its only argument place) have to be filled with predicates or propositions. Let's call this type of predicates *proposition operator* (cf. definition 19).

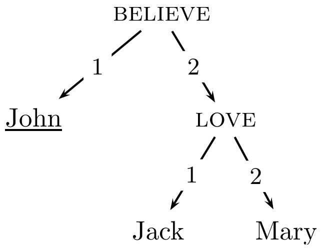
e) Examples 6 show that our metalanguage fails to represent important aspects of meanings: If example 6d represents the meaning of *Maybe, John believes that Jack loves Mary*, what is the representation for *John, who perhaps believes that Jack loves Mary.*? Unlike the first, the latter sentence does not refer to a state of affairs, but to an individual, namely John. Yet, this individual is described as a participant of a state of affair. Therefore, exactly the same proposition is involved in the meanings of the two sentences. We can solve this problem by introducing a new convention into our metalanguage: That part of a meaning that is supposed to be the *referring* element (or the semantic head) is underlined. (Cf. examples **??**)

Examples 7 P-SYNONYMY.

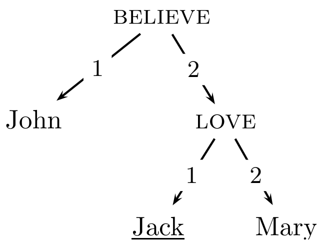
a) The meaning of *John believes that Jack loves Mary*



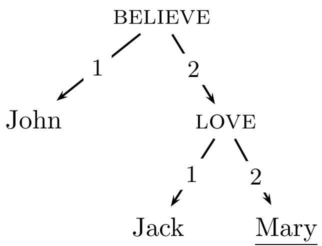
b) The meaning of *John who believes that Jack loves Mary*



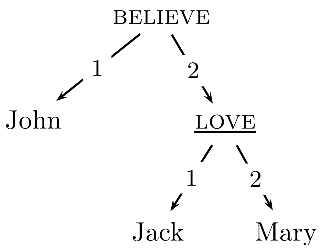
c) The meaning of *Jack, whom John believes to love Mary*



d) The meaning of *Mary, whom John believes to be loved by Jack*



e) The meaning of *Love, which John believes Jack feels for Mary*



Examples 7 show that there is another interesting type of synonymy, which is stronger than p-synonymy. We could call it r-synonymy (*reference* synonymy):

Definition 18 R-SYNONYMOUS. In this paper, two (predicative) expressions e_1 and e_2 are called r-synonymous iff they are p-synonymous and if they refer to the same entity (i.e. if in their meaning representation the same token is underlined).

Definition 19 PROPOSITION OPERATOR. In this paper, we call predicates which accept only predicates or propositions as arguments (in any of their argument places) proposition operators.

Definition 19 allows us to define a new, important concept: *simple sentence*.

Definition 20 SIMPLE SENTENCE. A simple sentence is an assertive sentence whose meaning doesn't contain any proposition operators.

Remark 2 SIMPLE SENTENCE. Any complex assertive sentence can be (recursively) broken up into (1 to n) simple sentences by deleting all proposition operators. (Junctors like *and* and *or* are also proposition operators. They may connect an unbounded number of simple or complex sentences.)

Definition 21 MAIN PREDICATE. Let S be a simple sentence (in the sense of definition 20). That predicate of S which is not argument of any other predicate is called in this paper the main predicate of S.

With these definitions we can formulate two important axioms:

Axiom 4 In order to study a language's lexemes, it is useful to study its simple sentences: They characterize the basic predicates and the basic semantic names of a language.

Axiom 5 In order to give a linguistically correct treatment to a sentence, one has to determine the simple sentences it consists of and the main predicates of the simple sentences.

Determining a sentence's main predicate presupposes knowing all predicative lexemes (with all their properties) of the language in question. As yet, noone has an exhaustive list of the predicative lexemes of any language. It is even worse: Noone has defined clear criteria for distinguishing predicative lexemes from semantic names. No linguist would doubt that there is such a thing as predicativity, but there is still no satisfying account of it. Definition 10 is, of course, not the answer. It is the question. The rest of this paper can be viewed as an attempt to improve definition 10.

Figure 2.6: A very short introduction into the basic ideas of model theoretic semantics including a model for relation no. 2 of figure 2.1

A model $\mathcal{M}$ for a vocabulary $\mathcal{V}$ is a pair $\langle \mathcal{U}, \mathcal{N} \rangle$, where $\mathcal{V}$ is a set of individual and predicate constants, $\mathcal{U}$ is a set of individuals and relations and $\mathcal{N}$ is a function from $\mathcal{V}$ to $\mathcal{U}$.

Let V be a vocabulary, i.e

- a set of individual constants $\{j, p, m, \ldots\}$
  (a representation of the words *John, Paul, Mary, ...*)
- a set of one-place predicate constants $\{sleeps, walks, is\_nice, \ldots\}$
  (a repr. of the expressions *sleep, walk, be nice, ...*)
- a set of two-place predicate constants $\{loves, hates, knows, \ldots\}$
  (a repr. of the expressions *love, hate, know ...*)
- ...

Let $M = \langle U, N \rangle$ be a model. Let U consist of

- a set of individuals: $\{\overline{j}, \overline{p}, \overline{m}, \ldots\}$
  (a repr. of the persons John, Paul, Mary, ...)
- a set of one-place relations or properties:
  - $\overline{sleeps} = \{\overline{m}, \overline{p}, \ldots\}$
    (a repr. of the fact that Mary sleeps, and the fact that Paul sleeps)
  - $\overline{walks} = \{\overline{j}, \ldots\}$
    (a repr. of the fact that John walks)
  - $\overline{is\_nice} = \{\overline{j}, \overline{p}, \overline{m}, \ldots\}$
    (a repr. of the fact that John, Paul and Mary are all nice)
  - ...
- a set of two-place relations:
  - $\overline{loves} = \{\langle j, m \rangle, \langle j, p \rangle\}$
    (a repr. of the facts that John loves Mary and Paul)
  - $\overline{hates} = \{\}$
    (a repr. of the fact that noone hates noone)
  - $\overline{knows} = \{\langle j, m \rangle, \langle j, p \rangle, \langle m, j \rangle, \langle m, p \rangle, \langle p, j \rangle\}$
    (a repr. of the fact that John, Mary and Paul all know each other with the exception that Paul doesn't know Mary)
- ...

Let N consist of

- $N(j) = \overline{j}$, $N(p) = \overline{p}$, $N(m) = \overline{m}$, ...
- $N(sleeps) = \overline{sleeps}$, ...
- $N(loves) = \overline{loves}$, ...
- ...

Rule 1 *An expression of the form $P_1(i)$, where P_1 is a one-place predicate and i is an individual constant, is true in a model $\mathcal{M}$ iff $N(i) \in N(P_1)$, i.e. iff $\overline{i} \in \overline{P_1}$. Comment: This rule matches sentences like* John is nice *and* Mary sleeps.

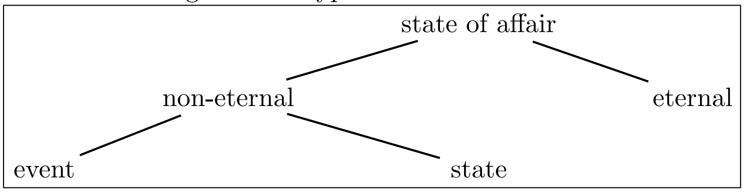
Rule 2 *An expression of the form $P_2(i, j)$, where P_2 is a two-place predicate and i and j are individual constants is true in a model $\mathcal{M}$ iff $\langle N(i), N(j) \rangle \in N(P_2)$, i.e. iff $\langle \overline{i}, \overline{j} \rangle \in \overline{P_2}$. Comment: This rule matches sentences like* John loves Mary *and* Mary knows Paul.

Rule 3 *An expression of the form $A \wedge B$ is true in a model $\mathcal{M}$ iff A is true and B is true in $\mathcal{M}$.*

Exercise 1 *Define a rule for expressions of the form $A \vee B$ and $\neg A$, i.e. for sentences like* John loves Mary and Paul knows John *or* Mary doesn't like Paul, *respectively.*

Exercise 2 *Determine wether the following sentences are true in M.* John loves Mary, Paul is intelligent. *Hint: If a state of affair is not in M, then it is not a fact according to M. Any model claims to be a model of the whole universe.*

Figure 2.7: Types of states of affairs



2.2 Types states of affairs

According to definitions 15 and 16, there is a correspondence between propositions and states of affairs or predicates and relations, respectively.

There are basically three different types of predicates (and therefore states of affairs). Some denote events (like WALK, MURDER or the zero-place predicate RAIN) and can therefore be modified by time and place predicates (like IN LONDON or AT MIDNIGHT). Others denote spatio-temporal *states* (like BE NICE, BE HEAVY, BE IN LOVE WITH). These can only under specific circumstances be modified by time and place predicates. (cf. *At that time, John was still quite nice.* vs. ?*John was nice at midnight.*) A third type of predicates denotes non-spatio-temporal states or relations (including one-place relations), like IDENTITY or PARITY. These predicates can never be modified by time or place predicates. (Cf. $*$ *At that time, 4 was even.*)

Ontologically, the different states of affairs can be characterized as follows: Events and spatio-temporal states are both entities that occur at some time and at some place. The difference is that events involve *change within short time intervals.* You can only said to be nice or intelligent, if you are nice or intelligent for some not too short time. You do not change quickly with respect to niceness or intelligence, though, in principle, you may cease to be nice or intelligent. Yet, if you eat an apple or kill somebody, you and the apple or the victim undergo change within a very short time interval. (Before the event, the apple or the victim is in good order, after the event, he has disappeared or is dead, respectively.) Non-spatio temporal states do not involve change at all: If 4 is even, it is always even. One may not be even at some time and odd at some other time. Unlike being nice, being even is an *eternal* property.

2.3 Argument structure

Predicates have a set of interesting properties, which we summarize with the term *argument structure.*

Definition 22 ARGUMENT STRUCTURE. The argument structure of predicate P is the set of the following properties of P:

a) number of arguments

b) semantic role of each argument

c) possible syntactic realizations of P

Consider the verb *sell.* Its meaning is the predicate represented in example 5a. According to our representation, SELL has four arguments: The seller-argument (1), the good-argument (2), the buyer-argument (3) and the price-argument (4). As we consider predicates to be their own 0th arguments, the total number of arguments is five.

With the names for the argument places, we have already introduced the notion of *semantic role.* A semantic role is, roughly, a set of implications of a predicate for one of its arguments. The predicate SELL, for instance, implies for its buyer-argument that it denotes a person or a set of persons[10]. If the first argument place of SELL is filled with an NP that does not denote a person, a semantically odd expression results: ?? *The gene has sold a protein*. We will not examin semantic roles further, because in our project we will not code information about them.

The *possible syntactic realizations* of a predicate P include basically two kinds of properties:

- The set of lexemes L_i which have (an aspect of) P as their meaning.
- The ways in which the L_i can be used in order to talk about P-states of affairs

Example: For the predicate OBSERVE there are at least the following lexemes L_i: *observe*, *observation*, *observer*. If you use one of these lexemes in a sentence, you speak — implicitly or explicitly — about an observation event. You may not use these lexemes in an arbitrary way. Each of them comes with certain rules about how to use them. Example: In order to say that John made some observations in Chinese verbs, you may utter example 8a or b, but not c.

Examples 8 SYNTACTIC REALIZATION OF A PREDICATE'S ARGUMENTS.

a) John made some observations in Chinese verbs.

b) John observed something in Chinese verbs.

c) ∗ John made some observations at Chinese verbs.

d) John machte einige Beobachtungen an chinesischen Verben.

This and similar phenomena cannot be explained with general (syntactic) rules. It is not for English syntax nor for syntax in general that example 8c is unwellformed. The German translation for *at* is in most cases *an*. In German, example 8d is well formed.

We explain the wellformedness of examples 8a and b by claiming that they are instances of certain *argument patterns* of the predicate OBSERVE. OBSERVE has the following argument structure:

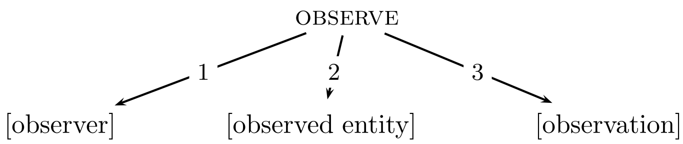


Figure 2.9 shows some argument patterns, i.e. ways of syntactically realizing the arguments of OBSERVE. Note that a predicate's (complete) set of argument patterns codes all aspects of syntactic realization. It codes the set of lexemes L_i which have the predicate as their meaning. And it codes all ways these lexemes can be used.

[10]Note that companies, cities, and so on, are sets of persons. Note that *sex* in *Sex sells* does not code the buyer argument of SELL. It codes the instrument operator: 'Sex is a good means for selling things'.

Figure 2.8: Some argument patterns of RESIST(1: [resisting entity], 2: [resisted entity])

	Pattern	*Lemma*	*Instance of the pattern*
1	(DetP_AP1_NNP0:_toNP2)NP	resistance	The wish to evoke *a Jewish resistance to Nazism* relates to a history which comprehends his own writings and example.
2	(NP0:_ofNP1)NP	resistance	Stalin did lead *the heroic resistance of the Soviet peoples* in the anti-Fascist war
3	(PossNP1_NP0:)NP	resistance	*Its resistance* is exemplified by the recent moving statues
4	(NPs1_VP:_NP2)S	resist	*De Valera himself resisted right-wing pressures*
5	(NP2_VP-encounter_ANP0:_fromNP1)NP	resistance	*Brazil encountered fierce resistance from Germany*
6	(AP1_NP0:_againstNP2)NP	resistance	Erich Honecker was locked up in March 1945 for organising *Communist resistance against the Nazis.*
7	(NP0:_ofNP1_toNP2)NP	resistance	no study is made of *the resistance of personalities to educational measures*
8	(N2_N0:_ofNP1)NP	resistance	Where the *fire resistance of a cast-iron member* must be increased
9	(NP1_VP-cop_AP0:_toNP2)S	resistant	*Chinese Yellow is very resistant to poor weather.*
10	(NP1_VP:_NP2)S	resist	*Germany fiercely resisted Brazil.*
11	(NP1_VP-offer_ANP0:_toNP2)SVK	resistance	*Germany offered fierce resistance to Brazil.*
12	(NP1_VP-show_ANP0:_toNP2)SVK	resistance	the 1937 Irish constitution, *which has ever since shown considerable resistance to reformulation.*
13	(NP2_VP-encounter_ANP0:_AP1_NNP0)SVK	resistance	*Brazil encountered fierce German resistance.*

Figure 2.9: Some argument patterns of OBSERVE

	Pattern	*Lemma*	*Instance of the pattern (taken from a corpus)*
1	`(NP1:)NP`	observer	some of which may never be apparent to *outside observers*
2	`(NP1:_ofNP2)NP`	observer	Leonard was *a very close observer of this scene*
3	`(NP0:)NP`	observation	Wölflinn's writings are strong on *observation*.
4	`(NP3:)NP`	observation	*This observation* has direct bearing on questions of authenticity.
5	`(NP0:_ofNP1)NP`	observation	You can rely upon *the observations of Sven Hjerson* for this. Sven Hjerson is always making observations.
6	`(NP0:_ofNP2)NP`	observation	basing his teaching practice on *the observation of nature*
7	`(NP3:_aboutNP2)NP`	observation	he made *an observation about the passage on Bernini's St Teresa*
8	`(NP0:_ofNP3)NP`	observation	All of *these observations of correlations between ERPs and different kinds of behaviour ...*
9	`(NP0:_ofNP2)NP`	observation	All of *these observations of correlations between ERPs and different kinds of behaviour ...*
10	`NP3:_ofNP1`	observation	We have also included *the observations of a leading pub designer* in order to give the report more breadth .
11	`(N2_N0:)NNP`	observation	To carry out *meteor observation*
12	`(PossNP1_ANP0:_thatS3)NP`	observation	This returns us to *Arendt's observation that secrecy is a prerequisite of totalitarianism*
13	`(N[instr]_N0:_ofNP2)NNP`	observation	*Video Observation of Meteors*
14	`(N[instr]_N2_N0:)NNP`	observation	Here is a list of recently published material dealing with *video meteor observation*:
15	`(NP1_VP0_NP2)S`	observe	*The writer will have closely observed the material discussed.*
16	`(NP2_VPP0)S`	observe	*A man was observed* on the steps of one of the banks in Dublin's College Green last week.
17	`(NP2_VPP0_byNP1)S`	observe	*This turn was observed by several witnesses.*
18	`(NP2_VPP0_byNP[instr])S`	observe	*The resulting cloud was observed by instruments on satellites and aircraft*
19	`(Inp0_NP2)Inp`	observe	He was told to *observe a fish in a tank.*
20	`(Inp0_NP3_amongstNP2)Inp`	observe	It is not surprising, therefore, to *observe a tendency amongst materialists to argue...*

Figure 2.10: Lexical categories used in the argument patterns

```
 1) n          -->  [noun]
 2) n_gen      -->  [genitive noun]            # hitchhiker's
 3) en         -->  [proper noun]
 4) en_gen     -->  [genitive proper noun]     # John's
 5) v          -->  [verb]
 6) adj        -->  [adjective]
 7) adv        -->  [adverb]
 8) det        -->  [determiner]
 9) possdet    -->  [possessive determiner]    # my, your, his, ...
10) pron       -->  [pronoun]
11) prep       -->  [preposition]
12) pres_part  -->  [present participle]       # walking, thinking ...
13) past_part  -->  [past participle]          # walked, thought
14) v_inf      -->  [infinitive form of a verb]
15) cnj        -->  [conjunction]
16) infp       -->  [infinitive particle]      # to
17) intj       -->  [interjection]
18) be         -->  [some forme of the verb 'to be']
```

2.3.1 Argument patterns

The syntactic categories used in the argument patterns of figure **??** are defined by the figures 2.10 and 2.11.

The meaning of the argument patterns will be discussed in greater detail in the sections of chapter (**??**), because the question how argument patterns should be defined can only be adequately discussed after having introduced Mel'čuk's Lexical Functions. In this section I will only give a short overview. Yet, even for a quick overview, some definitions are necessary.

Definition 23 N_0(X) $(i > 0)$. [11] "The event which X denotes or which the referent of X is involved in as participant." Examples:

a) N_0(Mörder) = Mord

N_0(murderer) = murder

N_0(asesino) = asesinato

b) N_0(ermorden) = Mord

N_0(murder) = murder

N_0(asesinar) = asesinato

Definition 24 N_i(X) $(i > 0)$. [12] "Entity that participates in an X-event (or -state) as participant number i". Examples:

a) N_1(Mord) = Mörder

N_1(murder) = murderer

N_1(asesinato) = asesino

b) N_2(schießen) = Ziel

N_2(shoot) = target

[11] Mel'čuk calls this function S_0. We use N instead of S in order to save S for *sentence*.

[12] Mel'čuk calls this function S_i. We use N instead of S in order to save S for *sentence*.

Figure 2.11: Syntactic categories used in the argument patterns

```
 1) NP         -->  DetP? ANP
 2) ANP        -->  AP? NNP
 3) NNP        -->  (n* n:)|en:
 4) NP         -->  pron:|en:|(det? AP? n* n:)|en:
 5) DetP       -->  det
 6) DetP       -->  PossNP
 7) AdvP       -->  adv* adv:
 8) AP         -->  AdvP? adj* adj:
 9) PossNP     -->  (det? ap? n_gen:)|en_gen:|possdet
10) Inp        -->  AdvP? v_inf: AdvP? (PP?|NP?|AP?|AdvP?)
11) Inp        -->  v_inf: AP AdvP? (NP|PP)?
12) VP         -->  AdvP v:
13) VP         -->  'will' AdvP v:
14) VP         -->  'will' have AdvP v:
15) VPP        -->  be AdvP past_part:
16) S          -->  NP v NP? PP*
17) Sqwh       -->  [wh-question]
18) Sqalt      -->  [alternative question]
19) Past_Part  -->  past_part
20) Pres_Part  -->  pres_part
21) VPing      -->  Pres_Part
```

N_2(tirar) = blanco

c) N_2(lehren) = Stoff

N_2(teach) = (subject) matter

N_2(enseñar) = materia

d) N_3(lehren) = Schüler

N_3(teach) = pupil

N_3(enseñar) = alumno

Definition 25 V_0(X). The verb which expresses the event or state which X denotes or which the referent of X is involved in. Examples:

a) V_0(Mord) = ermorden

V_0(murder) = murder

V_0(asesinato) = asesinar

b) V_0(Mörder) = ermorden

V_0(murderer) = murder

V_0(asesino) = asesinar

Definition 26 A_0(X). The adjective which expresses the event or state which X denotes or which the referent of X is involved in. Examples:

a) A_0(Widerstand) = $\emptyset$

A_0(resistance) = resistant

A_0(resistencia) = $\emptyset$

b) A_0(Eifersucht) = eifersüchtig

A_0(jealousy) = jealous

A_0(celos) = celoso

Now let us have a look at figure 2.8: Firstly, all the patterns have the form `(X)Y`. This means: The whole pattern describes a phrase of category `Y`. The categories of a pattern's subphrases are defined by the figures 2.10 and 2.11. The arabic numerals i in the patterns mean: "This category codes participant number i of the event in question". The colon means: "This category is the semantic head of the pattern". The head of a pattern's head category must unify with the morphological base form of value of the corresponding lexical function.

Some examples: Pattern number 3 of figure 2.8 is to be read in the following way: If there is a nominal which consists of a PossNP and an NP, and if the head of the NP equals $N_0(resist)$, then the PossNP codes (or might code) participant number 0 of the resistance event.

Some examples: Pattern number 3 of figure 2.8 is to be read in the following way: If there is a sentence which consists of an NP, a VP with the copula (*is*, *was*, *will be*, etc.) as head, an AP, the preposition *to* and an NP, and if the head of the AP unifies with (the baseform of) $A_0(resist)$, then the first NP codes participant no. 1 and the second NP codes participant no.2 of the resistance event.

The real coding of predicative lexemes is not done with patterns as given in figure 2.8, but with *meta-patterns*. Figure 2.12 gives some meta patterns over np-patterns. The idea is the following: The meta-patterns are matched syntactically, a concordance for each predicative lexeme is calculated and then the variables of the meta-patterns are replaced with the right numbers. This is done with the GUI shown in figures **??** and **??**.

2.3.2 Arguments vs. operators

In definition 10 I introduced the concept of *argument place*. A lexeme which fills a predicate's argument place is said to act as an argument of the predicate. But not all words or lexemes that depend — in some way or other — on a predicate are usually considered arguments of the predicate. Some of them are usually considered *operators* of the predicate, or, the predicate is considered to be an argument of them. Example: In the sentence *Last Saturday, John kissed Mary very softly in the garden* only *John* and *Mary* are arguments of the predicate KISS. The other words are usually either viewed as operators of KISS or as parts of operators of KISS or as operators of operators of KISS, e.g. like in figure 2.15.

Up to now, noone has been able to give an exhaustive and non-controversial list of tests to tell apart arguments from operators. And, which is even worse, it is by no means clear what types of argument and operator places there are. Yet, if one wants to code the argument structure of predicates, such tests and such a classification of types of, say, possible complements of predicates is necessary. Therefore I will try to give such a classification and formulate tests for distinguishing different types of arguments and operators.

But before we try to define such a little theory of types complements, let us try to show that it is really necessary. Some linguists seem to believe that it is completely impossible to do this. They use to suggest to simply treat all complements in the same way, e.g. to classify all of them as arguments. Why should this not be a good solution?

Several answers are possible. I would like to give only one very pragmatic argument: If we treat all of a predicate's "complements" (in the widest sense of the word), as arguments, then almost any predicate can in principle be ascribed a huge number of argument places. Consider:

a) <u>John</u> kissed <u>Mary</u>.

b) <u>John</u> kissed <u>Mary</u> <u>in the garden</u>.

c) <u>John</u>, <u>knowing that she doesn't like him very much</u> kissed <u>Mary</u> <u>in the garden</u>.

Figure 2.12: Some meta-np-patterns

pattern	*example*
`DetP~_AP-i~_NNP-x:`	the (Conservative)2 (confusion)0
`DetP~_AP-i~_NNP-x:_aboutNP-j~`	the (Conservative)2 (confusion)0 (about taxes)3
`DetP~_AP-i~_NNP-x:_ofNP-j_asAP-k~`	the (german)1 (perception)0 (of the issue)2 (as problematic)3
`DetP~_AP-i~_NNP-x:_ofNP-j_asNP-k~`	the (german)1 (perception)0 (of the issue)2 (as a problem)3
`DetP~_AP-i~_NNP-x:_ofNP-j~`	the (german)1 (perception)0 (of the issue)2 ; the (german)1 (perception)3 (of the issue)2
`N-i_N-x:`	to carry out (meteor)2 (observation)2
`NP-x:_aboutNP-i`	he made (a remark)0/3 (about the passage in the book)2
`NP-x:_againstNP-i`	the (match)0 (against Germany)1
`NP-x:_aroundNP-i`	the (debates)0 (around cultural diversity)2
`NP-x:_atNP-i`	(amazment)0 (at the book)0
`NP-x:_atSqwh-i`	(amazement)0 (at how much stuff I sold)2
`NP-x:_betweenNP-i`	(contact)0 (between Professors and students)1
`NP-x:_betweenNP-i_andNP-j`	(interaction)0 (between ANP)1 (and the renin-angiotension system)1
`NP-x:_byNP-i`	(the abandonement)0 (by my beautiful God)1
`NP-x:_byNP-i~_fromNP-j`	(the abstention)0 (by Jehova's Witnesses)1 (from voting)2
`NP-x:_byNP-i~_ofNP-j`	(the abandonement)0 (by circulating libraries)1 (of the three volume novel)2
`NP-x:_forNP-i`	the (application)0 (for the job)2
`NP-x:_fromNP-i`	(the abdication)0 (from all responsability)2
`NP-x:_fromNP-i_toNP-j`	(transmission)0 (from an infected mother)4 (to her baby)3
`NP-x:_inNP-i`	an (observation)3 (in animals)2 ; (a believer)1 (in Melcuk)2
`NP-x:_intoNP-i`	an (investigation)0 (into the case)2 ; (an inquiry)1 (into the meaning of meaning)2
`NP-x:_ofNP-i`	(the absence)0 (of a date)1 ; (the activation)0 (of the transcription)2 ; (the german equivalent)1 (of Bush)1 ; (a good observer)1 (of animals)2 ; (A former employee)2 (of BMW)1 ; (the observations)3 (of a leading pub designer)1 ; (the practice)2 (of artistic directors)1
`NP-x:_ofNP-i_byNP-j~`	(the activation)0 (of the gene)2 (by the protein)3
`NP-x:_ofNP-i_toInp-j~`	(the ability)0 (of polymerases)1 (to discriminate against errors)2

Figure 2.13: The GUI for coding predicative lexemes

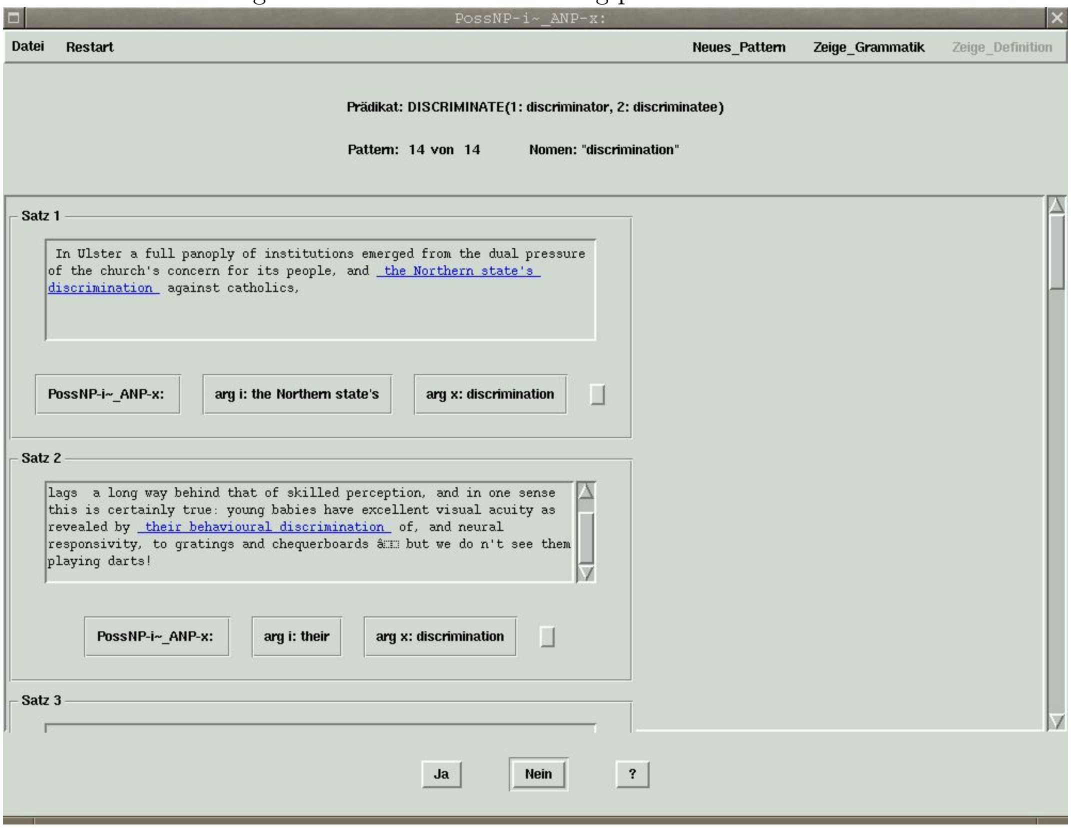


Figure 2.14: The GUI for coding predicative lexemes

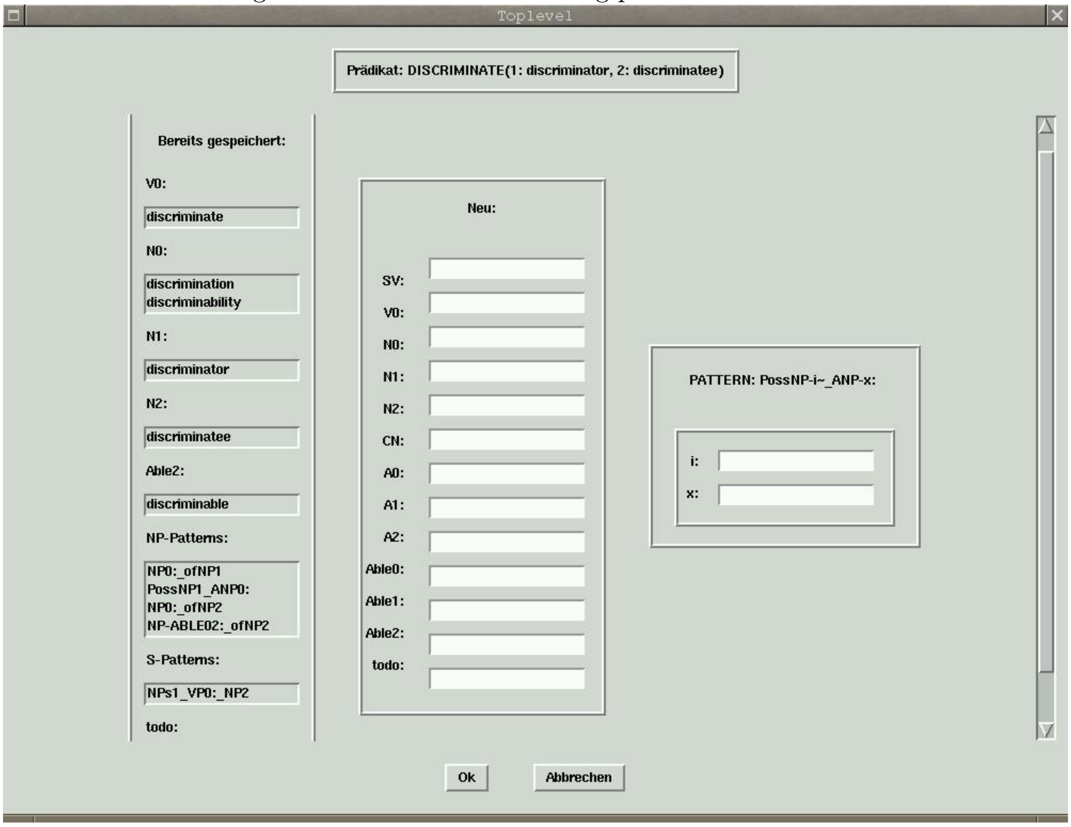


Figure 2.15: The proposition expressed by *Last saturday, John kissed Mary very softly in the garden*

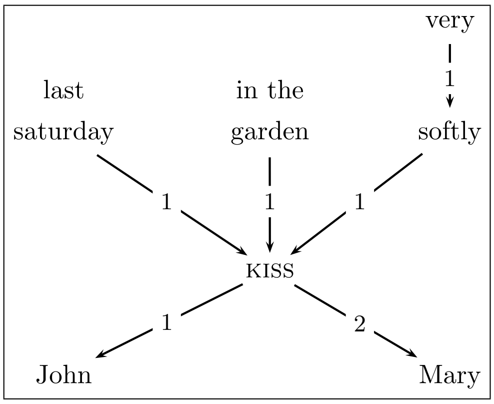


Figure 2.16: Types of predicate complements

	happened-test	presupposition test	that-test
argument	−	−	−
time-place event operator	+	−	−
general event operator	+	+	−
proposition operator	−	?	+

d) John, knowing that she doesn't like him very much, having talked to Jack about this issue, kissed Mary in the garden.

e) John, knowing that she doesn't like him very much, having talked to Jack about this issue kissed Mary in the garden while balancing a pencil on his nose.

f) ...

Yet, if we want to code argument places of predicates, we will have to start somewhere. Obviously, it would be nice to have a criterion of importance. If the number of complements is in principle unbounded, it would make sense to begin with the most important complements. The idea traditionally connected with (true) arguments is that this is exactly what they are: The most important complements of their operators.

One first idea how arguments could be distinguished from operators is to say that operators can be left away without producing an unwellformed expression. This criterion is well known and it is also well known that it doesn't really work: For any predicate, there are circumstances under which those complements which one would want to classify as arguments, can be left away. Consider e.g. second argument of EAT in the sentence *John is eating.* Although it does not have to be realized, most linguists tend to view it as a true argument, because in any eating event *something* has to be eaten, be it expressed or not. So, being obligatory is not a good criterion for being an argument.

But there are some other tests which can serve as a criterion for telling apart types of complements, if we use them *together*. These are: the happened-test, the presupposition-test and the that-test. I don't know who has first formulated each of them. They can be found in the linguistic literature in different forms, with different names and mostly without any references to previous formulations.

Figure 2.16 gives an overview about the types of complements in whose existence I tend to believe at the time being[13] and the test results which define them.

Definition 27 HAPPENED-TEST. Let P be a predicate. In order to use the happened-test, proceed as follows:

a) Find a predicative lexeme L which has P as its meaning and formulate a sentence S which contains the complement C to be tested. The test sentence has to be an active sentence in the past tense. (This is necessary to prevent effects of diathesis, aspect and the like.) Proceed as follows:

   1. Rules about C

      i. If C can be realized in S either by a nominal phrase or by a prepositional phrase, choose the prepositional phrase for building S. (Example: If P = GIVE, you might build sentences like S_1 and S_2: S_1= *John gave Mary the book.* S_2 = *John gave the book to Mary.* In this case, choose S_2, because it codes the reciever argument with a prepositional phrase.

[13] I use to often change my mind about them. If you have any qualms about figure 2.16, send them to `js@cis.uni-muenchen.de`

2. Rules about P

   i. If there is a verb V with $||V|| = P$, choose V to build the sentence.
   Example: If $P =$ LOVE, build a sentence with the verb *love* (and not, e.g. with the noun *love*).

   ii. If there is no verb V, such that $||V|| = P$, and if there is an adjective A or a common noun N, such that $||A|| = P$ or $||N|| = P$, build the sentence with A (or N) and the past tense copula ("was").
   Example: For the predicate BEAUTIFUL there is no verb. Build a sentence of the form *X was beautiful.*
   Example: For the predicate LINGUIST there is no verb, either. Build a sentence of the form *X is a linguist*[14].

   iii. If there is neither a verb V, such that $||V|| = P$, nor an adjective A or a noun N with the required meaning, try to find some complex verb CV (e.g. a support verb construction) such that $||CV|| = P$ and build the sentence with the complex verb. Example: For the predicate FATHER, there is no verb. (Like ** X fathers X.*) Choose a sentence of the form *X is the father of Y.*

b) Let S' be the sentence that results from your test sentence, if C is deleted from S — without any further modifications! Let PRO be a personal pronoun coreferring with the subject of S. If one of the following sentences is a) acceptable and b) a paraphrse of S, then the happened-test is positive:

   1. S'. This happened C.
   2. S'. PRO did it/this/so C.
   3. S'. PRO was C, when he did it.

Examples 9 HAPPENED-TEST. Consider the following test sentences. The complement to be tested is underlined.

a) <u>John</u> kissed Mary.
∗ Kissed Mary. This happened <u>John</u>.
∗ Kissed Mary. He did it <u>John</u>.
∗ Kissed Mary. He was <u>John</u>, when he did it.
This means: the happened-test is negative.

b) John kissed <u>Mary</u>.
∗ John kissed. This happened <u>Mary</u>.
∗ John kissed. He did it <u>Mary</u>.
∗ John kissed. He was <u>Mary</u>, when he did it.
This means: the happened-test is negative.

c) John kissed Mary <u>in the garden</u>.
John kissed Mary. This happened <u>in the garden</u>.
This means: the happened-test is positive.

d) John kissed Mary <u>softly</u>
John kissed Mary. He did it <u>softly</u>
This means: the happened-test is positive.

e) Zaphod left <u>the Vortex</u> unimpressed.
∗ Zaphod left unimpressed. This happened <u>the Vortex</u>.
∗ Zaphod left unimpressed. He did it <u>the Vortex</u>.
∗ Zaphod left unimpressed. He was <u>the Vortex</u>, when he did it.
This means: the happened-test is negative.

[14] *Linguist* is quite a special type of predicate. This type will be discussed below

f) Zaphod left the Vortex unimpressed.
Zaphod left the Vortex. He was unimpressed, when he did it.
This means: the happened-test is positive.

Remark 3 HAPPENED-TEST. The rationale of the happened-test is quite simple: It is used to distinguish true arguments from event operators. Event operators are predicates, which take events as their arguments. They either express properties of events, i.e. *in what way* some *happened* or they express properties of event participants, i.e. *how* they are involved in the event or *how* they *did* what they did. Obviously, it would make sense to define subclasses of general event operators: while *slowly* in *The door opened slowly* expresses a property of the door's opening, i.e. of the event, *unimpressed* in *Zaphod left the Vortex unimpressed* expresses a property of an event participant.

Definition 28 PRESUPPOSITION TEST. Let P be a predicate. In order to use the presupposition test, proceed as follows:

a) Find a predicative lexeme L which has P as its meaning and formulate a sentence S which contains the complement C to be tested. The test sentence has to be an active sentence in the past tense. (This is necessary to prevent effects of diathesis, aspect and the like.)

b) Let S' be the sentence that results from your test sentence, if C is deleted from S. If S' can be transformed in a way such that C's deletion doesn't lead to unwellformedness (for example by passivizing it), do so. Add a subordinate clause to S' which negates that a *specific* entity of $C's$ type be involved in the state of affairs expressed by S'.[15] If the resulting sentence is a) acceptable and if it b) makes sense, i.e. can possibly be true, then the presupposition test is positive.

Examples 10 PRESUPPOSITION-TEST. Consider the following test sentences. The complement to be tested is underlined.

a) John kissed Mary in the garden.
† Mary was kissed in the garden, but there was no specific entity who kissed her.
This means: The presupposition test is negative.

b) John kissed Mary in the garden.
† John kissed in the garden, but he didn't kiss a specific entity.
This means: The presupposition test is negative.

c) John kissed Mary in the garden.
† John kissed Mary, but he didn't do so in a specific place.
Comment: Any kissing event is a spatio-temporal event. It has to occur somewhere. *This means: The presupposition test is negative.*

d) John kissed Mary yesterday.
† John kissed Mary, but he didn't do so at a specific time.
Comment: Any kissing event is a spatio-temporal event. It has to occur at some time. *This means: The presupposition test is negative.*

e) John kissed Mary softly.
John kissed Mary, but he didn't do it in a specific way.
This means: The presupposition test is positive.

f) Probably, John kissed Mary.
? John kissed Mary, but he didn't do it with a specific probability.
Comment: The presupposition test results in '?', because it is unclear, if this sentence makes sense/can possibly be true. Though, prima facie, it doesn't seem to make much sense, there may be people who think it does.

[15] Unfortunately, I haven't found a possibility to make this definition more operational without thereby excluding interesting cases. But I think a look at examples **??** will help to understand the definition.

g) Necessarily, John kissed Mary.
? John kissed Mary, but he didn't do it with a specific degree of necessity or possibility.
*Comment: The presupposition test results in '?', because it is unclear, if this sentence makes sense/can possibly be true. Further, it is unclear what type of entity*necessity*is.*

Remark 4 PRESUPPOSITION TEST. The rationale of the presupposition test is quite simple: It is used to distinguish those properties of an event, which any event of the type t in question must have in order to be an event of type t: Any kissing event must have a kisser and a kissed entity. And any kissing event must occur at some time and at some place. But a kissing event does not have to occur in a certain way, e.g. softly, in order to be a kissing event. Occuring in a certain way is, so to say, a contingent (vs. essential) property of kissing events.

The presupposition test produces dubious results for proposition operators. This doesn't matter, because, proposition operators can be identified with the that-test.

Definition 29 THAT-TEST. Let P be a predicate. In order to use the that-test, proceed as follows:

Find a predicative lexeme L which has P as its meaning and formulate a sentence S which contains the complement C to be tested. The test sentence has to be an active sentence in the past tense. (This is necessary to prevent effects of diathesis, aspect and the like.) Delete C from S. Build a sentence S' of the following form:

$$\text{It is } C' \text{ that S}$$

, where C' is an adjective or adjective phrase such that $||C'|| = ||C||$. Note the *is*: You may not change it to *was* or the like. If you cannot find any C', use C instead. If S' is a) accepbtable and b) a paraphrase of S, then the that-test is positive.

Examples 11 THAT-TEST.

a) Probably, John kissed Mary.
It is probable that John kissed Mary.
This means: The that-test is positive.

b) Most probably, John kissed Mary.
It is most probable that John kissed Mary.
This means: The that-test is positive.

c) Maybe John kissed Mary.
It is possible that John kissed Mary.
This means: The that-test is positive. Comment: Of course, the C' does not have to be morphologically related to C.

d) Probably, John kissed Mary.
??? It is Mary that probably, John kissed.
This means: The that-test is negative.

e) John kissed Mary softly.
$*$ It is softly/soft that John kissed Mary.
This means: The that-test is negative.

f) John kissed Mary in the garden.
$*$ It is in the garden that John kissed Mary.

g) Of course, he came
It is clear that he came.

Remark 5 THAT-TEST. The rationale of the that-test is quite simple: It is used to recognize proposition operators, i.e. predicates which accept only predicates or propositions as arguments (in any of their argument places). Propositions can be always be expressed as sentences S. And operators O of propositions can always be expressed in the form O *that* S.

Exercise 1 COMPLEMENT TEST. COMPLEMENT TYPE. The following sentences contain quite a lot of complements. Determine their predicative structure in the following way: Determine the simple sentences they consist of. Apply the complement tests given above to each simple sentence.

a) Most probably, the government raised the tax from 5% by 5 points to 10%.

b) It is possible that John bought in Munich a car from Mary for €2000.

c) According to Zaphod, Slartibartfast planned the construction of Norway thouroughly.

Exercise 2 COMPLEMENT TEST. COMPLEMENT TYPE. The complement tests and the list of complement types given above are far from being perfect. One goal of the project is to improve them. Examples for problems:

a) John cuts Mary's hair.

b) John buys Mary a book.

Use the complement tests to determine the predicative structure of these sentences. Do you think that new complement types should be postulated? Try to find other sentences which show that the list of complement types and/or the complement tests should be improved. How should they be improved?