

KIT REPORT 106

Defaults in Machine Translation

Birte Schmitz
J. Joachim Quantz

TECHNISCHE UNIVERSITÄT BERLIN
PROJEKT KIT, FR 5-12
FRANKLINSTR. 28/29, W-1000 BERLIN 10

e-mail: {birte,jjq}@cs.tu-berlin.de

February 1993

Abstract

In this paper we try to combine experiences gained in the Machine Translation project *FAST* with research concerning the default extension of the terminological representation system *BACK*. We analyze various types of ambiguities that are problematic for translation and show that disambiguation is possible on the basis of default assumptions. Some of these defaults form preference rule systems, while others are merely an elegant way of modeling exceptions. We show how the ideas underlying the *FAST* implementation can be reformulated in a declarative framework by using terminological logics.

We present an approach towards disambiguation which we call interpretation as exception minimization. The basic idea is to compare the possible interpretations with respect to the defaults (preference rules) they violate. Given a relevance ordering between multisets of defaults, we chose the interpretation yielding the minimal number of exceptions with respect to this ordering as the preferred one.

Contents

1	Introduction	2
2	Cases of Ambiguity	3
2.1	Structural Ambiguity	5
2.2	Lexical Ambiguity	6
2.3	Referential Ambiguity	9
2.4	Intertwining of Different Kinds of Ambiguity	11
3	Types of Defaults in MT	12
4	Disambiguation with Defaults	14
4.1	Formal Text Representation	14
4.2	Terminological Logics	15
4.3	Text Representation with Terminological Logics	18
4.4	Defaults in Terminological Logics	19
4.5	Disambiguation as Exception Minimization	21
4.6	Nonmonotonic Reasoning	21
5	Conclusion	23
	References	24
A	Syntax and Semantics of BACK V5	28
B	A Preference Semantics for Defaults in BACK V5	31

1 Introduction

Ambiguity is an essential feature of natural language (NL) utterances: though each utterance is normally quite unambiguous in its utterance situation, abstraction from this situation makes the utterance and its constituents highly ambiguous. In natural language processing (NLP) only a limited amount of the utterance situation is explicitly available, and on the basis of this limited information adequate disambiguation is beyond the scope of most existing systems. Consequently, some systems are restricted to more or less unambiguous input whereas others “avoid” disambiguation by constructing multiple structures or structures that are themselves ambiguous. In a sense it might thus be concluded that the existing NLP systems are unable to process *natural* language.

One important source for this shortcoming is the restriction to strict information in the underlying formalisms. In unification based formalisms like HPSG, information is modeled by constraints which are interpreted as strict rules, i.e. rules without exceptions. But only a small amount of linguistic knowledge can be expressed by necessary and sufficient conditions. In most cases not even necessary conditions can be specified, but only defaults, i.e. rules that allow for exceptions.

In this paper we will therefore explore how disambiguation in the context of Machine Translation (MT) can be supported by a default formalism. We will proceed in two steps: we begin by listing typical cases of ambiguity and the criteria which we see as crucial for disambiguation. This presentation is based on results achieved in the *FAST* project, an MT project focusing on the interpretation of anaphora [Schmitz et al. 91]. We will then investigate how the default extension proposed for terminological logics in [Quantz, Royer 92] can be used to support the disambiguation tasks.

When presenting our criteria for disambiguation and their relationship to defaults, we will explicitly treat the following questions:

1. What exactly is a default in the chosen application?
2. Which inferences are expected to be drawn by the default system?
3. Why can't these inferences be obtained by using classical logic—or if they can, what is the advantage of using defaults instead of classical logics?

We think that these questiones should be considered by any paper dealing with the application of default theories. Up to now, application oriented papers using defaults merely denounce classical logics as unsuitable tools for formalization. Instead of detailed arguments for the advantages of defaults in the particular application there is often only a general claim for the necessity of default reasoning.

Papers on nonmonotonic reasoning, on the other hand, usually contain a general motivation for the need of nonmonotonic inference systems and then focus

on technical problems arising in the design of such systems. They proceed by discussing examples, more often than not involving Tweety, and appeal to the reader's intuition when discarding or favorizing inferences. Though this procedure is a necessary step in the development of reasoning systems, we feel the need for testing nonmonotonic theories in realistic applications. In this sense our paper aims at achieving a tighter coupling between an application-oriented investigation and a theoretical framework for default reasoning.

2 Cases of Ambiguity

In MT a distinction is often made between disambiguation within the source language and disambiguation with respect to the target language. Correspondingly the translation process is subdivided into analysis and transfer (and synthesis where disambiguation processes probably occur as well but shall not be considered here). An example for such a transfer model is given in Figure 1.¹

There is no natural cut between disambiguation within the source language and disambiguation with respect to the target language; it is more a pragmatic boundary resulting from the division of labor between analysis and transfer. This can be derived from the fact that it is not a priori clear what counts as ambiguous within the source language.² Several possibilities to translate a lexeme do not necessarily prove a corresponding ambiguity of this lexeme within the source language. The problem here is that the same state of affairs can be expressed quite differently in different languages. In particular, the information that has to be expressed explicitly differs from language to language. For translating the German sentence

(1) Er schwimmt im Pool.

into English

(2) He/It is floating/swimming in the pool.

we have to decide whether the filler of the agent role is swimming (actively moving his arms) or just floating. In German there is no need to express this difference.³ Although 'schwimmen' has two possible translations into English, it is usually not claimed to be an ambiguous lexeme within the German language.

¹For a more profound description of the figure cf. [Schmitz et al. 91]; GPSG is the abbreviation of *Generalized Phrase Structure Grammar* [Preuss 87]; the *Functor-Argument-Structure*, a formalism especially tailored for the needs of MT, is described in [Hauenschild, Umbach 88], [Busemann, Hauenschild 89]; the evaluation of hypotheses and disambiguation of anaphoric relations will be partly presented in the section about referential ambiguity.

²There are several tests proposed for distinguishing between vagueness and ambiguity (cf. [Hacken 90]) yielding different results, however.

³The difference can be expressed in German ('treiben/schwimmen'), if it is regarded as relevant.

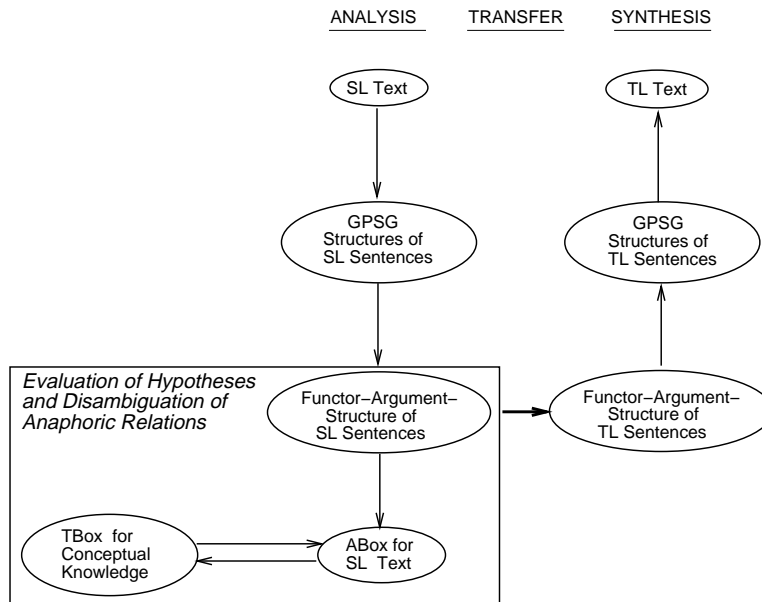


Figure 1: The MT Model of the Berlin Project *FAST*

These arguments support the idea of a variable depth of analysis. It seems to be the best translation strategy to disambiguate an expression only in cases where the information is needed for transfer. Preserving the ambiguity is probably the most efficient way of translating if the target language expresses the information in the same ambiguous manner, as in e.g.

(3) He saw the man with the telescope.

(4) Er sah den Mann mit dem Fernglas.

More crucial is the fact that disambiguation occurs at all stages of processing: at the syntactic as well as at the semantic and the conceptual level. Traditionally at least the following kinds of ambiguities are distinguished:

- **structural ambiguity.** Well known examples are sentence (3) above and:

(5) They're cooking apples.

(6) Every man loves a woman.

- **referential ambiguity,** like e.g.:

- (7) In its analysis of the shortcomings of Europe’s research potential, the Commission identified large gaps in the research continuum between universities on the one hand regarding work as too applied and industry on the other hand regarding it as too basic. (*Taken from the corpus “Proposal for a Council Decision Adopting the First European Strategic Programme for Research and Development in Information Technology” that ~~FAST~~ deals with, referred to as ESPRIT-Corpus*).

In this sentence all singular noun phrases (except ‘analysis’) are possible antecedents⁴ for ‘its’ and ‘it’.

- **lexical ambiguity**, as in

(8) His house is on the left bank.

(9) Peter has money in the bank.

A subtype of lexical ambiguity is categorial ambiguity like in ‘fish’ (noun or verb).

But all these different kinds of ambiguity reduce to the same scenario: There is more than one representation of an input phrase. On what basis can we select the preferred one? To get a better idea of this, let us have a closer look at several examples for each kind of ambiguity.

2.1 Structural Ambiguity

Due to the (semi-)free word order of German, there are two possible syntactic structures for sentence (10): either ‘die Hersteller’ is subject and ‘Schaltkreise’ is object or vice versa. To decide which representation is adequate, the system needs further information.

- (10) Die Hersteller produzieren Schaltkreise. (*translation of the preferred reading into English: The manufacturers produce integrated circuits.*)

The preferred reading is determined by making two assumptions:

1. If no other information is available, assume that the sentence is in unmarked word order. The subject of an unmarked sentence is topicalized.
2. The filler of the agent role usually is more “animate” than the filler of the affected role. For most active verbs the filler of the agent role is the subject.

⁴Antecedent and anaphor are used here to describe anaphorical as well as cataphorical ways of referring.

These assumptions can be seen as rules with exceptions (an exception to the first rule is given by example (11), exceptions to the second rule are the German sentence “Diktaturen produzieren Konformisten.” or the English sentence “This sort of environment produces criminal types.”). In sentence (10) both assumptions are in favor of the reading ‘die Hersteller’ = subject and ‘Schaltkreise’ = object. But consider the following German sentence:

- (11) Schaltkreise produzieren die Hersteller. (*Here the object is topicalized in the preferred reading. This can be expressed in English by means of passivization: Circuits are produced by the manufacturers.*)

In this case the assumptions are in conflict, since the first assumption favors ‘Schaltkreise’ to be subject, whereas the second advocates ‘die Hersteller’ as a subject. In order to get the adequate reading, we must make sure that assumption two has more influence than assumption one.

2.2 Lexical Ambiguity

Lexical disambiguation generally involves categorization as a subproblem. According to its semantic type a linguistic expression can be categorized (usually on the basis of a type hierarchy). Categorization is a special field of application for defaults. Semantic types are used to encode *selectional restrictions*. Selectional restrictions are a means to express semantic agreement between a functor and its arguments. They should ensure that not all words and phrases in the lexicon are combinable into more complex phrases. So, e.g. ‘die’ can only be applied to something that was once alive, i.e. the subject of ‘die’ should be ‘animate’. But even in technical reports there is a kind of metaphorical use of language. The following examples are discussed in [Kay et al. 91]:

- (12) The program died before it reached this routine.
 (13) The bill will probably die in the senate.
 (14) The movement died in the late thirties.

There are two general possibilities to formulate the selectional restrictions for ‘die’:

1. The agent of ‘die’ has to be an object that can undergo a change of state. This rule is applicable to a wide range of cases (including (12), (13), and (14)) and therefore its disambiguation effect is weak.
2. The agent of ‘die’ has to be an animate object. There are additional rules necessary to express the metonymical extension.

Another example is given by the large set of prepositions that can be used both temporally and locally ('in the seventies', 'in the pocket'). For these prepositions the analysis has to decide which reading is expressed in the input phrase. The semantic type of the core noun of the prepositional phrase can provide the necessary information in many cases. But obviously the semantic type cannot be seen as strict information. It is always possible to shift a type if forced by the context as in sentence (15).

(15) After Jalta the world looked different.

As we already stated, these kinds of type shifting do not only occur in poetic language. We rather see semantic type shifting (like ambiguity in general) as a universal means to ensure the efficiency of language. In the literature different cases of semantic type shifting⁵ are mentioned.

Firstly there are cases like

(16) John's dissertation, which weighs five pounds, has already been refuted. (*example due to* [Nunberg 78]).

(17) Plato, who is on the top shelf, was a great author. (*taken from* [Fauconnier 85]).

In these cases the semantics of the lexemes 'dissertation' and 'Plato' includes different aspects. The dissertation can be regarded both as a physical object with a weight and as its contents. 'Plato' can stand for the author or for his work. Since these lexemes can be interpreted without assuming special contexts we suggest that the different aspects and their relations should be covered by introducing one semantic type that includes all these aspects. With respect to such modeling no type shifting is involved.

Another case described as a kind of type shifting is demonstrated by the example:

(18) The mushroom omelet left without paying. (*From* [Fauconnier 85]).

Compared to the former examples, the context is much more important here. In a special context nearly any expression can be used to refer to any referent. The underlying principles are investigated by Fauconnier [Fauconnier 85]. His work is based on Nunberg's notion of *pragmatic function* (cf. [Nunberg 78]). A pragmatic function connects two domains (or mental spaces) in such a way that a description of an element of the first domain can be used to identify an element of the second domain. We have to examine which kind of information – either from the co-text or from background knowledge – contributes to finding the preferred interpretation. In example (18) the rule "'food' can be used to refer to a customer in the restaurant

⁵These cases cannot be distinguished by sharp boundaries.

domain” must be derived from the background knowledge. The verbal phrase ‘left without paying’ usually needs an human agent; this should lead to an interpretation of ‘mushroom omelet’ as a human agent. The system has to search for a suitable human object in a restricted domain.

Another quite promising approach to solve the problem of semantic type shifting is Pustejovsky’s proposal of *type coercion* (cf. [Pustejovsky 92], and for his general approach [Pustejovsky 91]). The idea is to regard lexemes as polymorphic functions. Pustejovsky imagines an expression being assigned a default typing and introduces the concept of type coercion as “a semantic operation that converts an argument to the type which is expected by a function, where it would otherwise result in a type error.” In addition to this, he states a *qualia structure* for nouns denoting objects consisting of four basic roles (cf. [Pustejovsky 91]). These roles comprise information about:

- the relation between an object and its parts/constituents,
- what distinguishes the object within a larger domain,
- the purpose and function of the object, and
- the factors involved in the origin of the object.

By means of type coercion together with the qualia structure Pustejovsky can explain why the sentence

(19) Mary enjoyed the book.

is usually interpreted as

(20) Mary enjoyed reading the book.

Lexical transfer usually involves disambiguation, as most lexemes give rise to more than one translation.

(21) He lived in a big apartment.

(22) He lived in the fifteenth century.

Both occurrences of ‘he lived’ can be translated into German ‘er lebte’, but in the first case (21) the translation ‘er wohnte’ is much more natural. The translation ‘er wohnte’ is not possible in the second case (22).

We suggest the following translation rules on the basis of the semantic type of the core nominal phrase in the prepositional phrase:⁶
 ‘live’ + prepositional phrase expressing a local extension → ‘wohnen’ (a strict

⁶Note what we already said about the semantic type as non-strict information.

über den See fahren	to cross the lake
mit dem Auto fahren	to drive to go by car
mit dem Fahrrad fahren	to cycle to go by bike
mit dem Aufzug fahren	to take the lift to ride the elevator
per Anhalter fahren	to hitchhike

Figure 2: Possibilities of translating ‘fahren’ into English.

rule),

otherwise: ‘live’ \rightsquigarrow ‘leben’ (a default).⁷

Another example is given by the possibilities of translating the German verb ‘fahren’ into English. Some are exemplified in Figure 2.2.⁸

It is necessary to have a default translation rule because in many cases the information that is needed for the correct choice, e.g. whether the agent of ‘fahren’ is the driver or just a passenger, or what kind of vehicle is used, is not explicitly contained in the clause itself, but possibly derivable from co-text or background knowledge. In these cases it is nevertheless necessary to get a preliminary translation on the basis of the most plausible hypothesis that might be revised if this hypothesis turns out to be wrong.

2.3 Referential Ambiguity

One task of analysis is to resolve inter- and intrasentential anaphoric references.⁹ This can only be done on the basis of a whole variety of different kinds of information. The following factors are relevant (Cf. [Hauenschild, Pause 83] and [Pause 86]):

- morphological factors (agreement in person, number and gender),
- syntactic factors (position and syntactic function of expressions in a sentence),

⁷It is quite obvious that this description is not fine-grained enough, since as sentence like ‘He lives in Munich’ is translated in ‘Er lebt in München’ although ‘in Munich’ expresses a local extension. This demonstrates the difficulty of finding adequate factors that guide translation. Empirical linguistic work on large corpora is necessary to get solid knowledge about the conditions of translation.

⁸Cf. also [Hauenschild 86].

⁹As far as they are relevant for translation.

- semantic factors (which thematic role is filled by a referent),
- thematic factors (what is thematic (topic, focus) in the sentence, in the text),
- factors concerning contents (what was already said about the referent),
- lexical factors (selectional restrictions, sense relations) and
- encyclopedic factors (which statements are compatible with each other on the basis of background knowledge).

In its current phase the Berlin MT Project *FAST* has been developing a method for the resolution of anaphoric personal and possessive pronouns (cf. [Schmitz et al. 91] and [Preuss et al. 92]).¹⁰ The following preference rules (with different weight) were implemented:

- An antecedent candidate which does not agree with the anaphor with respect to person, number and gender is regarded as very poor (negative preference).
- Antecedent candidates that do not fulfill the binding principles as defined in [Preuss et al. 92] are regarded as poor (negative preference).
- Structurally close antecedent candidates are preferred.
- The subject is preferred.
- The topic of the sentence is preferred.
- Free adjuncts are regarded as poor candidates (negative preference).
- Candidates that fill the same semantic role as the anaphor are preferred.
- The antecedent has to be compatible with the selectional restrictions that derive from the predications on the anaphor.

In order to get the best preferred antecedent from a list of candidates, the criteria are used to score each candidate. The scores are positive numbers for preference and negative numbers for negative preference of a candidate (depending on type of text, domain, and language). The scores are added up for each antecedent candidate and the candidate with the highest total score is taken to be the antecedent. The scores are set up according to the insight that criteria either reinforce each other or are in conflict. Conflicting criteria with the same weight lead to an ambiguity if no additional information is considered. Their scores are equal. So for example subject preference plus topic preference is in conflict with role identity in (23), and therefore they give the same score.

¹⁰Similar approaches are described in [Asher, Wada 88] and [Carbonell, Brown 88].

- (23) Industry regards the necessary work as basic research and the universities consider it as too applied.

In example (23), both ‘industry’ and ‘necessary work’ are structurally salient antecedent candidates for ‘it’ because ‘industry’ is both the subject and the topic of the preceding sentence whereas ‘necessary work’ has the same semantic role as ‘it’. The preference for ‘necessary work’ is based on complex conceptual knowledge (that we cannot represent yet).

2.4 Intertwining of Different Kinds of Ambiguity

Since an ambiguous expression usually does not occur in a completely unambiguous co-text, methods and criteria are required to resolve the combination of different kinds of ambiguity. In the following example lexical, referential and structural ambiguity is intertwined in the expression ‘its achievement’:¹¹

- (24) Mounting a “technology push” across the Community capable of achieving technical parity with, if not superiority over, our main competitors within the next ten years represents an ambitious objective that will require for its achievement a joint effort. (*Again from the ESPRIT-corpus.*)

referential ambiguity: The possessive pronoun ‘its’ can refer anaphorically to all singular noun phrases (except ‘achievement’ that is excluded by binding principles).

lexical ambiguity: ‘achievement’ has at least the following two readings:

- in the sense of “action of achieving”, in German ‘Erreichen’; ‘Erzielen’; ‘Erringen’; ‘Erlangen’.
- in the sense of “a thing done successfully”, in German ‘Leistung’; ‘Errungenschaft’.

structural ambiguity: In the expression ‘its achievement’ the pronoun can fill different roles:

- the agent role and
- the affected role.

How can these intertwined ambiguities be resolved? There is a clear structural preference for the subject of the clause in which the possessive pronoun occurs. In our case this is the relative pronoun ‘that’. ‘That’ refers back to ‘objective’.

¹¹This example is also discussed in [Hauenschild 91].

The next step is to test whether ‘objective’ can fill the role of ‘its’ in the complex nominal phrase ‘its achievement’.

The lexical and the structural ambiguity of ‘its achievement’ are interdependent: in the first reading in the sense of “action of achieving” the possessive pronoun can only express the affected role, whereas in the second reading in the sense of “a thing done successfully” ‘its’ has to fill the agent role. The conceptual knowledge should exclude ‘objective’ as a filler of the agent-role. But ‘objective’ is a reasonable filler of the affected role since the phrase ‘achieving an objective’ is a wide-spread collocation. Based on this, the system should derive the following results:

- ‘objective’ is the best antecedent candidate for the possessive pronoun ‘its’,
- the first reading of ‘achievement’ is the desired one,
- the possessive pronoun fills the affected role.

3 Types of Defaults in MT

The cases of ambiguity described above demonstrate that there are different fields of application of defaults. Based on these different fields we try to distinguish several types of defaults relevant for more satisfactory results in MT.

The first type is used in what we call (according to Jackendoff) *preference rule systems*. As we showed in Section 2 disambiguation relies on rather complex information in most cases. This complexity follows from the fact that a decision can only be made on the basis of a great number of competing and interacting factors. In order to demonstrate how these systems rule our decisions in daily life, Jackendoff gives the following examples:

Shall I buy the one I like, or the one that’s cheapest? Shall I answer the phone, or finish what I’m doing? Should I make more profit, or better preserve natural resources? In each case, two incommensurate preferences are in conflict, and one must determine a course of action that balances them. Of course, if the two preferences reinforce each other – the one I like best happens to be cheapest, answering the phone helps me finish what I’m doing, or the most profit can be made by maximally preserving natural resources, there is no difficulty in making a judgement. Thus these conscious preferences have the reinforcement-conflict patterns of preference rules. [Jackendoff 83, p.156]

Jackendoff sees the advantage of preference rule systems in their ability to derive “a quasi-determinate result from unreliable data”. A typical example of such a

system is given by the method and criteria for anaphora resolution as described in Section 2.

As preference rules are neither necessary nor sufficient conditions for an interpretation hypothesis, i.e. they are not strict rules, we propose to model them as defaults. Considering the explanatory power of preference rule systems the idea of stating the whole knowledge of the MT system in the form of preference rules suggests itself. Doing so would mean, however, to give up all advantages of strict knowledge (valid inferences, computability of subsumption etc.). This is why we aim at implementing a hybrid system with as much strict knowledge as possible and some additional default knowledge.

Intuitively an example like ‘fahren’ and its multiple translation possibilities is a type of default different from the type described so far (i.e. preference rule systems). This difference can be perceived by looking at the system from a procedural point of view: during run time these types of defaults are applied quite differently. In case of a preference rule system a list of possible solutions is given. Each member of this list instantiates the premise of each preference rule. In order to get the best candidate, an evaluation procedure is needed. This procedure has to test for each candidate whether the conclusions of each preference rule are true. The element that violates the qualitative minimal set of conclusions is considered to be the best one (see the next section for a more formal presentation).¹²

In cases like the ‘fahren’-example a conclusion is to be drawn on the basis of incomplete knowledge. This is done by a general rule. The derived conclusion might be overwritten by application of a more specific rule. Whereas the premises in a preference rule system usually are all of the same specificity, here there is a hierarchical ordering of the premises. The different degrees of specificity make a resolution of conflicting defaults feasible since a more specific default should win over a general one.

The type of default represented here by the ‘fahren’-example will be identified in the following as *defeasible inference rule*. From a procedural perspective the defeasible inference rules divide into two subtypes (with no sharp boundary between them). In the ‘fahren’-example the application of a general inference rule leads to the best solution with respect to a temporary state of the system. If this state changes, i.e. new information is added, the former “best solution” might be realized as poor. Therefore a kind of revision is required. Besides there are cases of defeasible inference rules that can be regarded as an elegant and efficient device to represent exceptions. E.g., the lexeme ‘high’ is usually translated into German ‘hoch’, ‘technology’ into German ‘Technologie’, but ‘high technology’ is translated as ‘Spitzentechnologie’. This can be modeled by the

¹²This process is quite similar to abductive reasoning. In abduction, from $p(X) \rightarrow q(X)$ and $q(a)$ it is concluded that $p(a)$. The additional device in a preference rule system has to find the best $p(a)$.

following translation rules:

high	\rightsquigarrow	hoch (default rule)
technology	\rightsquigarrow	Technologie (default rule)
high technology	\rightarrow	Spitzentechnologie (strict rule)

An application of defaults in the same fashion is discussed in detail in [Krieger, Nerbonne 91]. Krieger and Nerbonne propose a feature-based approach to represent the structure of words. In their approach defaults allow for the description of subregularities (i.e. “classes of items whose properties are largely but not perfectly regular”.) The authors state a paradigm for German modal verbs and propose to treat a suppletion like the German ‘sein’ via default overwriting.

These examples differ from the ‘fahren’-example with respect to the nonmonotonicity during run time, since they do not cause any revision. At any rate, either the specific information is available or no exception is at issue, and the general rule is applied.

4 Disambiguation with Defaults

In this section we will turn our attention towards a formalization of the ideas presented in the preceding sections. We sketch a purely declarative framework for text representation and indicate why the need for defaults arises. We hasten to add, however, that this reformulation within a logical formalism is not carried out on a detailed level yet (see [Quantz 93] for a more detailed presentation). Instead we aim at establishing a general connection between Machine Translation and Default Reasoning.

4.1 Formal Text Representation

The basic idea of formal text representation is to explicitly represent the information which is (implicitly) contained in a text. Consider, for example, sentence 10 again:

(10) Die Hersteller produzieren Schaltkreise.

In describing this sentence, or more precisely its preferred reading, we said that ‘Die Hersteller’ is its subject, is topicalized, is the filler of the agent role of ‘produzieren’ etc. Thus a formal representation of the preferred reading has to contain this information explicitly. Usually, this is realized by having a formal language with an entailment relation \models , such that a formal representation r entails all this information i , i.e. $r \models i$. Given the task of Machine Translation we are interested in a formal representation for a text containing at least the information necessary for determining its translation in the target language.

In the last decade unification-based feature formalisms such as GPSG [Gazdar et al. 85], LFG [Kaplan, Bresnan 82], or HPSG [Pollard, Sag 87] have become more and more popular as formalisms for text representation. Their basic idea is to use feature structures to formally represent the information contained in NL expressions. A feature structure is basically a set of feature-value pairs or, more formally, a partial function from features into values.

On a syntactic level, examples for features and appropriate values are case: {nom, gen, dat, acc} or number: {sing, plu} etc. Note that the value for a feature is not necessarily an atomic value but can be a complex feature structure itself, e.g. the value for a feature such as ‘subject’. Though most of the existing formalisms concentrate on the syntactic and, to a lesser degree, on the semantic level, it is in principle possible to encode arbitrary information, e.g. conceptual or encyclopedic information within this format.

Thus the general idea within the paradigm of unification-based feature formalisms is to use feature structures for formally representing information contained in NL texts. The obvious question from a computational point of view is then: how can feature structures be algorithmically constructed from NL texts.

In order to address this question we will formalize the task of interpretation within the framework of terminological logics (TL). Terminological logics are very similar to the feature logics underlying the unification-based formalisms from computational linguistics (see [Nebel, Smolka 87], [Backofen et al. 90], and [Carpenter 92]). They evolved from Semantic Networks and Frames, two representation formats widely used in Knowledge Representation, and are characterized by their logical foundation. A number of TL systems, such as BACK [Hoppe et al. 93], CLASSIC [Brachman et al. 91], or LOOM [MacGregor 91] have been developed during the 80s and are already used in NLP systems (e.g. LOOM in the Penman project [Bateman 92] and BACK in the project *FAST* [Schmitz et al. 91]).

4.2 Terminological Logics

In TL one typically distinguishes between *terms* and *objects* as basic language entities from which three kinds of formulae can be formed: *definitions*, *descriptions*, and *rules* (see the sample modeling below). A definition has the form $t_n \doteq t$ and expresses the fact that the name t_n is used as an abbreviation for the term t . A list of such definitions is often called *terminology*, hence the name Terminological Logics. All TLs provide two types of terms, namely *concepts* (unary predicates) and *roles* (binary predicates), but they differ with respect to the term-forming operators they contain. Common concept-forming operators are: conjunction ($c_1 \sqcap c_2$), disjunction ($c_1 \sqcup c_2$), and negation ($\neg c$), as well as quantified restrictions [Quantz 92] such as value restrictions ($\forall r:c$), which stipulate that all fillers for a role r must be of type c , or number restrictions ($\geq n r:c$ or $\leq n r:c$, stipulating that there are at least or at most n role-fillers of type c for r). Role-forming opera-

tors are, besides conjunction, disjunction, and negation, role composition ($r_1.r_2$), transitive closure (r^+), inverse roles (r^{-1}) and domain or range restrictions ($c|r$ or $r|c$). In a description, an object is described as being an instance of a concept ($o :: c$), or as being related to another object by a role ($o_1 :: r:o_2$). Rules have the form $c_1 \rightarrow c_2$ and stipulate that each instance of the concept c_1 is also an instance of the concept c_2 . A formal syntax and semantics for the TL implemented in BACK V5 is given in the appendix (Section A).

There are several ways in which TL systems have been employed in NLP. For one thing, they are used to store the contents of NL texts. Thus, the discourse referents associated with the referring expressions are modeled as TL-objects and the semantic contents of the expressions are used to describe these objects. To make such descriptions possible, the semantic contents of words have thus to be modeled as TL-concepts. Since most NL systems are restricted to texts about a specific domain, this kind of modeling is often called *domain modeling*.

Consider the highly simplified domain modeling below whose net representation is shown in Figure 3. One role and five concepts are defined, out of which four are primitive (only necessary, but no sufficient conditions are given). Furthermore, the modeling contains one rule and four object descriptions.

product	:<	\top
chemical product	:<	product
biological product	:<	product \sqcap \neg chemical product
company	:<	≥ 1 produces:product
produces	:<	domain (company)
chemical company	:=	company \sqcap \forall produces:chemical product
\exists produces:chemical product	\rightarrow	high risk company
toxipharm	::	chemical product
biograin	::	biological product
chemoplant	::	chemical company
toxiplant	::	≤ 1 produces \sqcap produces:toxipharm

In Tls, such a modeling is regarded as a set of formulae Γ . Given the formal semantics of a TL, such a set of formulae will entail other formulae, i.e., there is an entailment relation $\Gamma \models \gamma$. Now the service provided by TL systems is basically to answer queries whether some formula γ is entailed by a modeling Γ . The following list contains examples for the types of queries that can be answered by a TL system like BACK:

- $\Gamma \models t_1 \sqsubseteq t_2$
Is a term t_1 more specific than a term t_2 , i.e., is t_1 *subsumed* by t_2 ? In the sample modeling, the concept ‘chemical company’ is subsumed by ‘high risk company’, i.e., every chemical company is a high risk company.

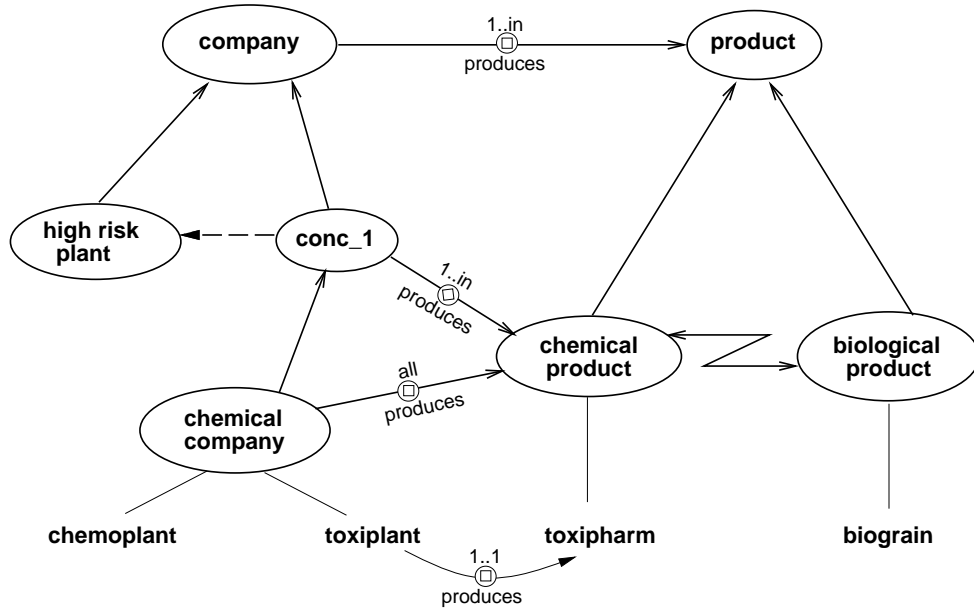


Figure 3: The net representation of the sample domain. ‘conc_1’ is the concept $\exists \text{ produces:chemical product}$.

- $\Gamma \models t_1 \sqcap t_2 \sqsubseteq \perp$
Are two terms t_1 and t_2 incompatible or disjoint? In the sample modeling, the concepts ‘chemical product’ and ‘biological product’ are disjoint, i.e., no object can be both a chemical and a biological product.
- $\Gamma \models o :: c$
Is an object o an instance of concept c (object classification)? In the sample modeling, ‘toxiplant’ is recognized as a ‘chemical company’.
- $\Gamma \models o_1 :: r:o_2$
Are two objects o_1, o_2 related by a role r , i.e., is o_2 a role-filler for r at o_1 ? In the sample modeling, ‘toxipharm’ is a role-filler for the role ‘produces’ at ‘toxiplant’.
- $\Gamma \models X :: c$
Which objects are instances of a concept c (retrieval)? In the sample modeling, ‘chemoplant’ and ‘toxiplant’ are retrieved as instances of the concept ‘high risk plant’.
- $\Gamma \cup \{\alpha\} \models \perp$
Is a description α inconsistent with the modeling (consistency check)?

With respect to the sample modeling, the description $\text{chemoplant} :: \text{produces:biograin}$ is inconsistent, i.e., ‘biograin’ cannot be produced by ‘chemoplant’.

4.3 Text Representation with Terminological Logics

It is straightforward to encode feature structures in terms of terminological logics: a feature structure can be encoded by a TL concept, where the features are encoded as TL roles. One important difference between the feature formalisms and TL concerns the distinction between objects and concepts. In TL, a description $o :: c$ consists of an object o , which corresponds to an FOL constant, and a concept c , which corresponds to a unary FOL predicate. The objects in TL are related by roles, which correspond to binary FOL predicates (thus $o_1 :: r:o_2$ corresponds to $R(o_1, o_2)$). We thus have a strict distinction between objects and types in TL.

The feature formalisms, on the other hand, do not make this distinction: in a sense, they lack the level of objects and only comprise types. Thus, the fillers of features are not objects, but feature structures, i.e. types. Instead of distinguishing between objects and types they distinguish between atomic and complex types: atomic types are values like ‘sing’ or ‘acc’, whereas complex types have an internal structure like ‘per:3’ etc. Note that there is no strict correspondance between atomic values and TL objects. Though each atomic value can be encoded as a TL object, the converse is not true: TL objects are structured because they can be described by structured concepts.

As a consequence of this difference between TL and feature formalisms, we need to introduce additional TL objects to encode complex feature structures. Thus the feature structure in 25 will be encoded by the TL formulae in 26.

$$(25) \left[f_1 \left[f_2 \ v \right] \right]$$

$$(26) \begin{array}{l} o_1 :: f_1:o_2 \\ o_2 :: f_2:v \end{array}$$

Thus we explicitly introduce names for the objects which are described by the feature structures (and by the internally nested feature structures).

Let us illustrate this encoding with a TL-representation of sentence 10:

$$(27) \begin{array}{l} e :: \text{phrase} \sqcap \text{phon:“Die Hersteller produzieren Schaltkreise”} \sqcap \\ \quad 4 \text{ atom} \sqcap \text{atom: } (w_1 \sqcap w_2 \sqcap w_3 \sqcap w_4) \\ w_1 :: \text{word} \sqcap \text{pos:1} \sqcap \text{phon:“Die”} \\ w_2 :: \text{word} \sqcap \text{pos:2} \sqcap \text{phon:“Hersteller”} \\ w_3 :: \text{word} \sqcap \text{pos:3} \sqcap \text{phon:“produzieren”} \\ w_4 :: \text{word} \sqcap \text{pos:4} \sqcap \text{phon:“Schaltkreise”} \end{array}$$

We see 27 as representing the information explicitly given by sentence 10. Since it contains mainly phonological information we use Π_e to denote this set of TL-formulae.

After this excursion we can readdress the problem of interpretation: how can we compute the information implicitly given by 27? Given the above formal framework, this can be formalized as computing an interpretation i_d such that

$$\Gamma \cup \Pi_e \models e :: i_d$$

i.e., given our grammatical knowledge Γ plus the phonological information Π_e , what additional information about e is entailed? Since i_d is deduced from Γ and Π_e this can be called the *deductive approach* to interpretation.

In the following we will not go into any details about the formulae contained in Γ , i.e., we will not deal with the exact formalization of grammatical information in TL (see [Quantz 93] for more details). We will rather concentrate on the general place of defaults within our framework. Given that Γ contains the appropriate formulae concerning German grammar, we expect the following entailment

$$\Gamma \cup \Pi_e \models e :: \text{subject} : p_1 \sqcup p_3$$

where p_1 is the phrase ‘Die Hersteller’ and p_3 is the phrase ‘Schaltkreise’. Thus the strict information in Γ is not sufficient to deduce whether p_1 or p_3 is the subject. The idea of having additional default information is to achieve the following entailment

$$\Gamma \cup \Pi_e \models_{\Delta} e :: \text{subject} : p_1$$

where \models_{Δ} is a nonmonotonic entailment relation taking into account the defaults in Δ .

4.4 Defaults in Terminological Logics

We will now present briefly the integration of defaults into terminological logics as proposed by Quantz and Royer in [Quantz, Royer 92]. Note that all types of information supported in traditional TL, i.e. definitions, rules, and descriptions are interpreted as strict. Defaults are integrated into this framework as rules that allow for exceptions. A default δ is thus represented as $c_1 \rightsquigarrow c_2$, where c_1 is called the premise of δ (δ_p) and c_2 the conclusion of δ (δ_c). A default δ can then be paraphrased as “each instance of δ_p is a δ_c unless it is an exception”. The obvious strategy for reasoning with defaults is to minimize exceptions. This is reflected in the preference semantics for defaults, as specified in [Quantz, Royer 92].¹³ Roughly speaking, a model M_1 is preferred over a model M_2 , iff it contains fewer exceptions than M_2 .¹⁴

¹³We include a summary of the formal definitions in the appendix (Section B).

¹⁴For the idea of preference semantics see [Shoham 88].

One of the problems arising in default theories is the treatment of conflicting information. Whereas in classical logics conflicting information leads to contradictions, conflicts in default logics arise from the possibility of exceptions and should therefore not entail contradictions. Several strategies for dealing with conflicting defaults have been proposed in the literature (cf., for example, [Brewka 91]).

In [Quantz, Royer 92] conflicting defaults result in partial skepticism, more precisely in disjunctive conclusions: suppose object o is an instance of both c_1 and c_2 , and there are two defaults $c_1 \rightsquigarrow c_3$ and $c_2 \rightsquigarrow c_4$; if these defaults conflict, i.e., if $c_3 \sqcap c_4 \sqsubseteq \perp$, then it is only inferred that o is an instance of c_3 or c_4 , but not of both. Furthermore, some conflicts between defaults can be resolved via a precedence ordering on defaults. Roughly speaking, if two defaults δ_1 and δ_2 conflict where δ_1 precedes δ_2 , then δ_2 is cancelled. The precedence ordering is induced by the subsumption hierarchy of the terminology: a default δ_1 precedes a default δ_2 if δ_1 is more specific, i.e., if $\delta_{1p} \sqsubset \delta_{2p}$. (The user can refine this ordering by specifying additional precedences $\delta_1 \prec \delta_2$.)

Before we investigate, how this default extension can support disambiguation in Machine Translation, we would like to point out some characteristics of defaults, especially with respect to nonmonotonicity. Nonmonotonicity in the above framework means that it is possible that $\Gamma' \not\models_{\Delta} \alpha$ and $\Gamma \models_{\Delta} \alpha$ even though $\Gamma \subset \Gamma'$. Roughly speaking, additional information can invalidate derived information. There are three aspects in which this nonmonotonicity is relevant for TL:

- In TL concepts form a subsumption hierarchy. This hierarchy can be seen as a graph (see Figure 3) and from this point of view it is often said that information is *inherited* from a node to its descendants. Whereas all strict information is inherited in this way, default information can be “overwritten” at subsumed nodes. If no multiple inheritance of defaults is involved, this aspect of nonmonotonicity is comparatively harmless. (Compare the use of defaults as an elegant way to model exceptions as indicated in Section 3.)
- To determine whether an object o is an instance of a concept c , the explicit descriptions of o are completed, i.e., implicit consequences of these descriptions are computed. This can be seen as deriving a canonical normal form, the most specific description for o . Without defaults, this description can be constructed incrementally simply by adding information. Due to the nonmonotonic character of defaults, however, it can become necessary to revise this description in the course of its construction.
- A similar revision problem occurs when descriptions are added to a set of TL-formulae. The additional information can invalidate descriptions previously derived at objects.

4.5 Disambiguation as Exception Minimization

We will now show how the default extension of TL can be used to formalize the disambiguation task. In Section 2 we have presented a number of examples for various types of ambiguity and have argued that the preferred interpretation is determined by preference rules. In addition to the encoding of strict grammatical information as TL-formulae we encode the preference rules as TL-defaults. Now experiences with anaphora resolution in the *FAST* project have shown that in general each interpretation will satisfy some of these preference rules while violating others. This suggests that we need a priority ordering relating multisets of defaults. Multisets, since an interpretation can contain several exceptions to the same default (e.g., two occurrences of ‘he’ which both are exceptions to the same default on anaphora).

Given such an ordering, we say that an interpretation i_1 is preferred over i_2 iff $E_1 \sqsubset E_2$, where E_i denotes the multiset of defaults to which i_i is an exception. The easiest way to obtain such an ordering is to assign to each default δ a natural number p_δ representing the price of violating it: the more important the default, the higher the price. We then define $E_1 \sqsubset E_2$ iff $\Sigma(E_1) > \Sigma(E_2)$, where $\Sigma(E_i) \stackrel{\text{def}}{=} \Sigma\{p_\delta : \delta \in E_i\}$. Note that this ordering can be partial only, which would explain truly ambiguous expressions in which neither $E_1 \sqsubset E_2$ nor $E_2 \sqsubset E_1$.

Since disambiguation yields the interpretation with the qualitatively minimal exceptions (according to the partial order), we call this approach *interpretation as exception minimization*.

To integrate this into the Quantz/Royer extension of TL, we need to expand the precedence relation on the set of defaults to a precedence relation on the powerset of defaults. Given such a precedence relation we can change the definition of exception preference [Quantz, Royer 92, Def. 5] (see Definition 4 in the appendix) accordingly, i.e., $M_1 \sqsubset_E M_2$ iff $E_1^2 \sqsubset E_2^1$.

Note that this formalization is essentially a declarative reformulation of the scoring device implemented in the *FAST* system.

4.6 Nonmonotonic Reasoning

Having sketched the formal framework of text representation and interpretation as exception minimization, we will now look at its impact on nonmonotonic reasoning (NMR). In the preceding sections we spoke of defaults as rules which allow for exceptions or as preference rules. This view is perfectly in line with the underlying motivation of NMR: “(m)uch of our experience of the world is available in the form of general rules which are not universally true; they may have exceptions but they express what is true under normal conditions” [Brewka 91, p. 6].

In the literature on NMR a number of different proposals can be found to integrate such defaults syntactically and semantically into formal logics. We will

not review these proposals here (see [Brewka 91] for an excellent overview), but rather indicate what inferential behavior we expect from defaults in our application.

First, we need a formalism in which both strict and defeasible information can be encoded. As Kilbury points out in [Kilbury 93] some nonmonotonic inheritance systems such as DATR [Evans, Gazdar 89] are deficient in the sense that they do not support the representation of strict information. Though we are not claiming that there is a clearcut a priori distinction between essential and accidental properties, we think that for pragmatic reasons it is advantageous to decide for a given domain which constraints are to be modeled as strict and which as defeasible.¹⁵

Second, the defeasible constraints have different weight. For one thing, feature-structures form a subsumption hierarchy and therefore more specific constraints should “overwrite” more general constraints. In addition to this specificity precedence, the user should be able to specify other precedences between defaults as well. For most nonmonotonic logics versions have been proposed which take into account priorities between defaults, e.g. Konolige’s hierarchic autoepistemic logic, Brewka’s cumulative default logic, or Lifschitz’s prioritized circumscription (see [Brewka 91] for details).

In all these approaches, however, priorities can only be expressed between single defaults whereas in our application we need to specify priorities between multisets of defaults. Whereas it is obvious how to extend Quantz/Royer’s notion of exception preference to integrate such a priority ordering (see above), the corresponding extension of the nonmonotonic logics mentioned above seems less straightforward.

In a sense we could modify Reiter’s Default Logic by specifying a preference ordering on the various extensions generated by a set of strict formulae and a set of defaults. We could define the notion of exceptions with respect to these extensions and then define the ordering similar to the exception preference by Quantz/Royer. One important difference between Reiter’s Default Logic and Quantz/Royer’s TL-extension, however, is that Reiter’s defaults are syntactic forward chaining rules, whereas Quantz/Royer’s defaults behave like material implications, i.e. contraposition and case-based inferences are valid under certain conditions (see [Quantz, Royer 92] for details). We think that this is appropriate in our application though this issue has to be evaluated more carefully.

Third, we are primarily interested in using defaults for deriving information about individuals. Thus we need the default entailment of descriptions $\Gamma \models_{\Delta} o :: c$, rather than of subsumptions $\Gamma \models_{\Delta} t_2 \sqsubseteq t_1$. There is a tendency in the literature to “homogenize” the formulae of nonmonotonic logics (see, for example, [Brewka 93] and [Dix, Schmitt 93]). Though we think that this is very useful for the theoretical characterization of nonmonotonic logics, we don’t see a strong need

¹⁵For the problem of processing purely defeasible information see also Brachman’s polemic [Brachman 85].

for doing so from an applicational point of view. We rather think it is advantageous to have a clear distinction between strict and defeasible information, and to use the defeasible information for deriving additional information about individuals, namely for determining the preferred interpretation of a given utterance.

5 Conclusion

We have discussed a number of examples demonstrating the ubiquity of ambiguities in natural languages. We have shown that a special type of defaults, namely preference rules, provides a suitable tool for the disambiguation of these examples. In contrast to the use of defaults as defeasible inference rules, which is also needed in Machine Translation as an elegant way to model exceptions, preference rules function allow selection between the multiple interpretations stemming from ambiguity.

We have sketched a declarative framework for disambiguation using a default extension of terminological logics. The basic idea is to choose the interpretation that induces the fewest exceptions to the preference rules encoded as defaults. This approach is therefore called interpretation as exception minimization.

A closer investigation of the interaction between defaults as defeasible inference rules and defaults as preference rules has to reveal whether an efficient and cognitively adequate strategy for exception minimization can be specified. We see the declarative framework presented here as a formal foundation for developing NL systems. To obtain efficient performance, however, we think that heuristics have to be integrated to choose the preferred interpretation. Thus instead of evaluating all possible interpretations by comparing the exceptions they induce, heuristics are needed to test as few interpretations as possible.

Acknowledgements

We would like to thank Bob Carpenter, Christa Hauenschild, Jim Kilbury, Susanne Preuß, Carla Umbach, and Emil Weydert for detailed and valuable comments on earlier drafts of this paper.

References

- [Asher, Wada 88] N. Asher, H. Wada, “A Computational Account of Syntactic, Semantic and Discourse Principles for Anaphora Resolution”, in *Journal of Semantics* **6**, 309–344, 1988
- [Backofen et al. 90] R. Backofen, H. Trost, H. Uszkoreit, *Linking Typed Feature Formalisms and Terminological Knowledge Representation Languages in Natural Language Front-Ends*, DFKI Research-Report 91–28, 1991
- [Bateman 92] J. Bateman, “The Use and Design of Linguistically Motivated Ontologies for Controlling and Deployment of Linguistic Resources”, in [Preuss, Schmitz 92], 50–99
- [Batori, Weber 86] I. Bátori, H.J. Weber (eds), *Neue Ansätze in maschineller Sprachübersetzung: Wissensrepräsentation und Textbezug*, Tübingen: Niemeyer, 1986
- [Bläsius et al. 87] K.H. Bläsius, U. Hedtstück, C.-R. Rollinger (eds), *Sorts and Types in Artificial Intelligence*, Berlin: Springer, 1987
- [Brachman 85] R.J. Brachman, “I lied about the trees or Defaults and Definitions in Knowledge Representation”, *The AI Magazine* **6**, 80–93, 1985
- [Brachman et al. 91] R. Brachman, D.L. McGuinness, P.F. Patel-Schneider, L. Alperin Resnick, A. Borgida, “Living with CLASSIC”, in [Sowa 91]
- [Bresnan 82] J. Bresnan (Ed.), *The Mental Representation of Grammatical Relations*, Cambridge (Mass): MIT Press, 1982
- [Brewka 91] G. Brewka, *Nonmonotonic Reasoning: Logical Foundations of Commonsense* Cambridge: Cambridge University Press, 1991
- [Brewka 93] G. Brewka, *A Framework for Cumulative Default Logics: Preliminary Report*, submitted for publication
- [Busemann, Hauenschild 89] S. Busemann, C. Hauenschild. “From FAS Representation to GPSG Structures”, in S. Busemann, C. Hauenschild, C. Umbach (eds), *Views of the Syntax/Semantics Interface*, KIT-Report 74, 1989
- [Carbonell, Brown 88] J.G. Carbonell, R.D. Brown, “Anaphora Resolution: a Multi-Strategy Approach”, in *Proc. of COLING 1988*, Vol. 1, Budapest, 96–1101
- [Carpenter 92] B. Carpenter, *The Logic of Typed Feature Structures*, Cambridge: Cambridge University Press, 1992

- [Dix, Schmitt 93] J. Dix, P.H. Schmitt, *Preferential and Rational Closure: A Comparative Study*, submitted for publication
- [Evans, Gazdar 89] R. Evans, G. Gazdar, "Inference in DATR", *ACL'89*, 66–71, 1989
- [Fauconnier 85] G. Fauconnier, *Mental Spaces: Aspects of Meaning Construction in Natural Language*, Cambridge, Ma.:Bradford, 1985.
- [Gazdar et al. 85] G. Gazdar, E. Klein, G.K. Pullum, I.A. Sag, *Generalized Phrase Structure Grammar*, Cambridge: Blackwell, 1985
- [Hacken 90] Pius ten Hacken, *Reading Distinctions in Machine Translation*, Working Papers in Natural Language Processing 6, Leuven/Utrecht, 1990
- [Hauenschild 86] C. Hauenschild, "KIT/Nasev oder die Problematik des Transfers bei der Maschinellen Übersetzung", in: [Batori, Weber 86], 167–195
- [Hauenschild 91] Ch. Hauenschild, "Anapherninterpretation in der Maschinellen Übersetzung" in *Zeitschrift für Literaturwissenschaft und Linguistik*, LiLi **84**, 50 – 66.
- [Hauenschild, Pause 83] Ch. Hauenschild, P.E. Pause "Faktoren-Analyse zur Modellierung des Textverstehens" in *Linguistische Berichte* **88**, 101–121, 1983
- [Hauenschild, Umbach 88] Ch. Hauenschild, C. Umbach. "Funktorkomplexstruktur. Die satzsemantische Repräsentations- und Transferenebene im Projekt KIT-FAST", in J.Schütz (ed), *Workshop Semantik und Transfer*, EUROTRA-D Working Papers 6, Saarbrücken, 16–35, 1988
- [Hobbs, Kameyama 90] J.R. Hobbs, M. Kameyama, "Translation by Abduction", in *COLING'90*, 155–161
- [Hoppe et al. 93] T. Hoppe, C. Kindermann, J.J. Quantz, A. Schmiedel, M. Fischer, *BACK V5 Tutorial & Manual*, KIT Report 100, Technische Universität Berlin, 1993
- [Jackendoff 83] R. Jackendoff, *Semantics and Cognition*, Cambridge: MIT Press, 1983
- [Kaplan, Bresnan 82] R.J. Kaplan, J. Bresnan, "Lexical Function Grammar", in [Bresnan 82]
- [Kay et al. 91] M. Kay, J.M. Gawron, P. Norvig, *VerbMobil: A Translation System for Face-to-Face Dialog*, August 1991

- [Kilbury 93] J. Kilbury, *Strict Inheritance and the Taxonomy of Lexical Types in DATR*, submitted for publication,
- [Krieger, Nerbonne 91] H.-U. Krieger, J. Nerbonne, *Feature-Based Inheritance Networks for Computational Lexicons*, DFKI Research Report 91-31, 1991
- [MacGregor 91] R. MacGregor, “Using a Description Classifier to Enhance Deductive Inference”, in *Proceedings Seventh IEEE Conference on AI Applications*, Miami, Florida, February, 1991, 141–147
- [Nebel, Smolka 87] B. Nebel, G. Smolka, “Representation and Reasoning with Attributive Descriptions”, in [Bläsius et al. 87], 112–139
- [Nebel et al. 92] B. Nebel, C. Rich, W. Swartout, *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR'92)*, San Mateo: Morgan Kaufmann, 1992
- [Nunberg 78] G. Nunberg, *The Pragmatics of Reference*, Bloomington, Indiana: Indiana University Linguistics Club, 1978.
- [Pause 86] P.E. Pause, “Zur Modellierung des Übersetzungsprozesses”, in [Batori, Weber 86]
- [Pearce, Wagner 92] D. Pearce, G. Wagner (eds), *Logics in AI, Proceedings of JELIA'92*, Berlin: Springer, 1992
- [Pollard, Sag 87] C. Pollard, I.A. Sag, *An Information Based Syntax and Semantics*, Vol. I Fundamentals, Stanford: CSLI Lecture Notes 13, 1987
- [Preuss 87] Susanne Preuß, *GPSG-Syntax für ein Fragment des Deutschen KIT-IAB 20*, TU Berlin, 1987.
- [Preuss, Schmitz 92] S. Preuß, B. Schmitz (eds), *Workshop on Text Representation and Domain Modelling*, KIT-Report 97, Berlin, 1992
- [Preuss et al. 92] S. Preuß, B. Schmitz, C. Hauenschild, *Anaphora Resolution Based on Semantic and Conceptual Knowledge* in [Preuss, Schmitz 92], 1–13
- [Pustejovsky 91] J. Pustejovsky, “The Generative Lexicon”, in *Computational Linguistics* **17**, 409–441, 1991
- [Pustejovsky 92] J. Pustejovsky, “Type Coercion and Lexical Selection” in J. Pustejovsky (Ed.), *Semantics and the Lexicon*, to appear
- [Quantz 92] J.J. Quantz, “How to Fit Generalized Quantifiers into Terminological Logics”, *ECAI-92*, 543–547

- [Quantz 93] J.J. Quantz, *Interpretation as Exception Minimization*, submitted for publication
- [Quantz, Royer 92] J.J. Quantz, V. Royer, “A Preference Semantics for Defaults in Terminological Logics”, in [Nebel et al. 92], 294–305
- [Royer, Quantz 92] V. Royer, J.J. Quantz, “Deriving Inference Rules for Terminological Logics”, in [Pearce, Wagner 92], 84–105
- [Schmitz et al. 91] B. Schmitz, S. Preuß, C. Hauenschild, *Textrepräsentation und Hintergrundwissen in KIT-FAST*, KIT-Report 93, 1991
- [Schmolze, Israel 83] J. Schmolze, D. Israel, “KLONE: Semantics and Classification”, in *Research in Knowledge Representation and Natural Language Understanding*, BBN Annual Report 5421, 27–39 1983
- [Shoham 88] Y. Shoham, *Reasoning about Change: Time and Causation from the Standpoint of Artificial Intelligence*, Cambridge: MIT Press, 1988
- [Sowa 91] J. Sowa (Ed.), *Principles of Semantic Networks: Explorations in the Representation of Knowledge*, 1991

A Syntax and Semantics of BACK V5

The terminological logic of BACK V5 (see [Hoppe et al. 93] for details) is specified by the following syntax (t_p are primitive terms, t_n term names):

$$\begin{aligned}
c &\rightarrow \top, \perp, c_p, c_n, \neg c_p, c_1 \sqcap c_2, \forall r:c_1, \geq n r, \leq n r, r:o \\
r &\rightarrow \perp, r_p, r_n, \neg r_p, r_1 \sqcap r_2, r_1^{-1}, r_1.r_2, c|r_1, r_1|c \\
\gamma &\rightarrow t_1 \sqsubseteq t_2, o :: c
\end{aligned}$$

For this language a modeltheoretic semantics can be given where a model M of a set of TL-formulae Γ is a pair $\langle D, \mathcal{I} \rangle$. \mathcal{I} maps concepts into subsets of D , roles into subsets of $D \times D$, and object-names injectively into D , in accordance with the following equations (we use $r(d)$ to denote $\{e : \langle d, e \rangle \in r\}$):

$$\llbracket \top \rrbracket^{\mathcal{I}} = D \quad (1)$$

$$\llbracket \perp \rrbracket^{\mathcal{I}} = \emptyset \quad (2)$$

$$\llbracket \neg t_p \rrbracket^{\mathcal{I}} = D \setminus \llbracket t_p \rrbracket^{\mathcal{I}} \quad (3)$$

$$\llbracket t_1 \sqcap t_2 \rrbracket^{\mathcal{I}} = \llbracket t_1 \rrbracket^{\mathcal{I}} \cap \llbracket t_2 \rrbracket^{\mathcal{I}} \quad (4)$$

$$\llbracket \forall r : c \rrbracket^{\mathcal{I}} = \{d \in D : \llbracket r \rrbracket^{\mathcal{I}}(d) \subseteq \llbracket c \rrbracket^{\mathcal{I}}\} \quad (5)$$

$$\llbracket \geq n r \rrbracket^{\mathcal{I}} = \{d \in D : |\llbracket r \rrbracket^{\mathcal{I}}(d)| \geq n\} \quad (6)$$

$$\llbracket \leq n r \rrbracket^{\mathcal{I}} = \{d \in D : |\llbracket r \rrbracket^{\mathcal{I}}(d)| \leq n\} \quad (7)$$

$$\llbracket r : o \rrbracket^{\mathcal{I}} = \{d \in D : \llbracket o \rrbracket^{\mathcal{I}} \in \llbracket r \rrbracket^{\mathcal{I}}(d)\} \quad (8)$$

$$\llbracket \{o_1, \dots, o_n\} \rrbracket^{\mathcal{I}} = \{\llbracket o_1 \rrbracket^{\mathcal{I}}, \dots, \llbracket o_n \rrbracket^{\mathcal{I}}\} \quad (9)$$

$$\llbracket r_1^{-1} \rrbracket^{\mathcal{I}} = \{\langle d, e \rangle \in D \times D : \langle e, d \rangle \in \llbracket r_1 \rrbracket^{\mathcal{I}}\} \quad (10)$$

$$\llbracket r_1.r_2 \rrbracket^{\mathcal{I}} = \llbracket r_1 \rrbracket^{\mathcal{I}} \circ \llbracket r_2 \rrbracket^{\mathcal{I}} \quad (11)$$

$$\llbracket c|r \rrbracket^{\mathcal{I}} = \llbracket r \rrbracket^{\mathcal{I}} \cap (\llbracket c \rrbracket^{\mathcal{I}} \times D) \quad (12)$$

$$\llbracket r|c \rrbracket^{\mathcal{I}} = \llbracket r \rrbracket^{\mathcal{I}} \cap (D \times \llbracket c \rrbracket^{\mathcal{I}}) \quad (13)$$

Satisfaction of formulae is then defined as follows:

$$M \models t_2 \sqsubseteq t_1 \quad \text{iff} \quad \llbracket t_2 \rrbracket^{\mathcal{I}} \subseteq \llbracket t_1 \rrbracket^{\mathcal{I}} \quad (14)$$

$$M \models o :: c \quad \text{iff} \quad \llbracket o \rrbracket^{\mathcal{I}} \in \llbracket c \rrbracket^{\mathcal{I}} \quad (15)$$

A structure M is a model of a formula γ iff $M \models \gamma$; it is a model of a set of formulae Γ iff it is a model of every formula in Γ . A formula γ is entailed by a set of formulae Γ (written $\Gamma \models \gamma$) iff every structure which is a model of Γ is also a model of γ .

Note that the following operators and formulae can be defined in this language:

$$nr \stackrel{\text{def}}{=} \geq nr \sqcap \leq nr \quad (16)$$

$$\exists r : c \stackrel{\text{def}}{=} \geq 1r|c \quad (17)$$

$$\exists r \stackrel{\text{def}}{=} \geq 1r \quad (18)$$

$$\neg \exists r : c \stackrel{\text{def}}{=} \leq 0r|c \quad (19)$$

$$\neg \exists r \stackrel{\text{def}}{=} \leq 0r \quad (20)$$

$$\geq nr : c \stackrel{\text{def}}{=} \geq nr|c \quad (21)$$

$$\leq nr : c \stackrel{\text{def}}{=} \leq nr|c \quad (22)$$

$$nr : c \stackrel{\text{def}}{=} nr|c \quad (23)$$

$$t_n \doteq t \stackrel{\text{def}}{=} t_n \sqsubseteq t, t \sqsubseteq t_n \quad (24)$$

$$c_1 \rightarrow c_2 \stackrel{\text{def}}{=} c_1 \sqsubseteq c_2 \quad (25)$$

Clearly, this logic is a subset of First-Order Logic with Equality. We can make this precise by specifying a translation of TL-formulae into FOL-formulae (cf. also [Royer, Quantz 92] and [Schmolze, Israel 83]). The set of concept-names and primitive concept components (resp. role-names and primitive role components) is bijectively mapped into a set of unary predicate symbols (resp. binary predicate symbols); object-names are bijectively mapped into constants. Let j be the corresponding bijection. We define the translation function $\bar{\gamma}$ as follows: a concept-name (or a primitive concept component) c is translated into $\lambda x C(x)$, where C denotes the unary predicate symbol representing c ($C = j(c)$); similarly, a role-name or primitive role component r is translated into $\lambda x \lambda y R(x, y)$, where R denotes the binary predicate symbol representing r ($R = j(r)$); object-names are translated into constants. We then define translation of arbitrary terms and TL-formulae:

$$\overline{\top} \stackrel{\text{def}}{=} \lambda x (\text{True}) \quad (\text{True is as special propositional letter}) \quad (26)$$

$$\overline{\perp} \stackrel{\text{def}}{=} \lambda x (\text{False}) \quad (\text{False is a special propositional letter}) \quad (27)$$

$$\overline{\neg c_p} \stackrel{\text{def}}{=} \lambda x (\neg C_p(x)) \quad (28)$$

$$\overline{c_1 \sqcap c_2} \stackrel{\text{def}}{=} \lambda x (\overline{c_1}(x) \wedge \overline{c_2}(x)) \quad (29)$$

$$\overline{\forall r : c} \stackrel{\text{def}}{=} \lambda x (\forall y \overline{r}(x, y) \rightarrow \overline{c}(y)) \quad (30)$$

$$\overline{\geq nr} \stackrel{\text{def}}{=} \lambda x (\exists y_1 \dots y_n \overline{r}(x, y_1) \wedge \overline{c}(y_1) \dots \overline{r}(x, y_n) \wedge \overline{c}(y_n) \wedge \mathbf{diff}(y_1, \dots, y_n)) \quad (31)$$

$$\overline{\leq nr} \stackrel{\text{def}}{=} \lambda x (\exists y_1 \dots y_{n+1} \overline{r}(x, y_1) \wedge \overline{c}(y_1) \dots \overline{r}(x, y_{n+1}) \wedge \overline{c}(y_{n+1}) \rightarrow \mathbf{eq}(y_1, \dots, y_{n+1})) \quad (32)$$

$$\overline{r : o} \stackrel{\text{def}}{=} \lambda x (\overline{r}(x, \overline{o})) \quad (33)$$

$$\overline{\{o_1, \dots, o_n\}} \stackrel{\text{def}}{=} \lambda x (x = \overline{o_1} \vee \dots \vee x = \overline{o_n}) \quad (34)$$

$$\overline{\neg r_p} \stackrel{\text{def}}{=} \lambda x \lambda y (\neg R_p(x, y)) \quad (35)$$

$$\overline{r_1 \sqcap r_2} \stackrel{\text{def}}{=} \lambda x \lambda y (\overline{r_1}(x, y) \wedge \overline{r_2}(x, y)) \quad (36)$$

$$\overline{r^{-1}} \stackrel{\text{def}}{=} \lambda x \lambda y (\overline{r}(y, x)) \quad (37)$$

$$\overline{r_1.r_2} \stackrel{\text{def}}{=} \lambda x \lambda y (\exists z \overline{r_1}(x, z) \wedge \overline{r_2}(z, y)) \quad (38)$$

$$\overline{c|r} \stackrel{\text{def}}{=} \lambda x \lambda y (\overline{r}(x, y) \rightarrow \overline{c}(x)) \quad (39)$$

$$\overline{r|c} \stackrel{\text{def}}{=} \lambda x \lambda y (\overline{r}(x, y) \rightarrow \overline{c}(y)) \quad (40)$$

$$\overline{c_1 \sqsubseteq c_2} \stackrel{\text{def}}{=} \forall x (\overline{c_1}(x) \rightarrow \overline{c_2}(x)) \quad (41)$$

$$\overline{r_1 \sqsubseteq r_2} \stackrel{\text{def}}{=} \forall x \forall y (\overline{r_1}(x, y) \rightarrow \overline{r_2}(x, y)) \quad (42)$$

$$\overline{o :: c} \stackrel{\text{def}}{=} \overline{c}(\overline{o}) \quad (43)$$

The formula $\text{diff}(y_1, \dots, y_n)$ states that the y_i are mutually distinct, the formula $\text{eq}(y_1, \dots, y_{n+1})$ that y_{n+1} is equal to one of y_1, \dots, y_n . Note that, by beta-reduction, the lambda-abstractions are eliminated from the translation of TL-formulae. Thus TL-formulae are translated into pure FOL-formulae.

B A Preference Semantics for Defaults in BACK V5

This is a simplified version of the preference semantics developed in [Quantz, Royer 92] (we dropped the notion of fulfillment-preference to keep the presentation simpler).

Definition 1 A default δ has the form $c_1 \rightsquigarrow c_2$ where c_1 and c_2 are arbitrary concepts. We say that c_1 is the **premise** of δ (written δ_p) and that c_2 is the **conclusion** of δ (written δ_c).

To resolve conflicts between defaults we define a precedence relation between defaults stemming from the hierarchical structure induced by definitions and rules:

Definition 2 Let Γ be any set of TL-formulae and δ_1, δ_2 any defaults. We say that δ_1 **precedes** δ_2 wrt Γ (written $\delta_2 \prec_\Gamma \delta_1$) iff $\Gamma \models \delta_{1p} \sqsubseteq \delta_{2p} \wedge \Gamma \not\models \delta_{2p} \sqsubseteq \delta_{1p}$.

The basic idea of preference between models consists in minimization of exceptions. We therefore define exceptions and compare exceptions in models:

Definition 3 Let δ be any default and M any structure.

The set of **exceptions** to δ in M is defined as

$$E_M(\delta) \stackrel{\text{def}}{=} \{d \in D : d \in \llbracket \delta_p \rrbracket^{\mathcal{I}} \wedge d \notin \llbracket \delta_c \rrbracket^{\mathcal{I}}\}.$$

Let M_1 and M_2 be any models. The set of **malus pairs** of M_1 with respect to M_2 is defined as

$$E_2^1 \stackrel{\text{def}}{=} \{\langle \delta, d \rangle : d \in E_{M_1}(\delta) \setminus E_{M_2}(\delta)\}.$$

Now a model is preferred if every exception in it is justified in the sense that it is an exception at a preceded default.

Definition 4 Let Γ be any set of TL-formulae, M_1 and M_2 any models of Γ , and Δ any set of defaults. M_2 is **Δ -preferred** to M_1 wrt Γ (written $M_1 \sqsubset_\Delta M_2$) iff

1. $E_1^2 \neq E_2^1$
2. $\forall \langle \delta, d_1 \rangle \in E_1^2 \exists \langle \delta', d_2 \rangle \in E_2^1 : \delta \prec_\Gamma \delta'$.

Finally, we define Δ -entailment as a preferential entailment in Shoham's sense:

Definition 5 A model M_1 is **Δ -preferred** iff there is no model M_2 such that $M_1 \sqsubset_\Delta M_2$. A set of formulae Γ **preferentially entails** γ (written $\Gamma \models_\Delta \gamma$) iff all Δ -preferred models of Γ are models of γ .