Gefördert vom
bmb+f
Bundesministerium für
Bildung, Wissenschaft,
Forschung und Technologie

Verbmobil
Verbundvorhaben

Semantic-Head Based Resolution of Scopal Ambiguities

Björn Gambäck, Johan Bos

Universität des Saarlandes,
Computerlinguistik

Report 194
Mai 1997

Mai 1997

Björn Gambäck, Johan Bos

Computerlinguistik
Bau 17.2
Universität des Saarlandes
66041 Saarbrücken

Tel.: (0681) 302 - 4680
Fax: (0681) 302 - 4351
e-mail: bos/gam@coli.uni-sb.de

Gehört zum Antragsabschnitt: 8

Die vorliegende Arbeit wurde im Rahmen des Verbundvorhabens Verbmobil vom Bundesministerium für Bildung, Wissenschaft, Forschung und Technologie (BMBF) unter dem Förderkennzeichen 01 IV 101 R gefördert. Die Verantwortung für den Inhalt dieser Arbeit liegt bei den Autoren.

Abstract

We present an algorithm for scope resolution in underspecified semantic representations. Scope preferences are suggested on the basis of semantic argument structure. The major novelty of this approach is that, on the one side, we maintain an (with respect to scopal ambiguities) underspecified semantic representation, and on the other side, at the same time suggest a scope resolution possibility. The algorithm is evaluated on four "real-life" dialogues and fares surprisingly well: about 80% of the utterances containing scopal ambiguities are correctly interpreted by the suggested resolution, leaving errors in only 5.7% of the overall utterances.

1 Introduction

Scopal ambiguities are known to be problematic for natural language processing (NLP). Resolving scope might lead to the well known combinatorial explosion puzzle. In NLP-applications like semantic-based machine translation, resolution of scope can be avoided if the translation takes place at a semantic representational level that encodes scopal ambiguites: the idea is to have a common representation for *all* of the possible interpretations of an ambiguous expression, as in [ACGR91]. Scopal ambiguities in the source language can then be carried over to the target language. Recent research [Rey93, Bos95, Pin95, CCvE$^+$96, Poe95] has used the term *underspecification* to describe this idea.[1]

However, underspecification has a clear disadvantage. Syntactic restrictions on scope ambiguities are not encoded in underspecified representations. This lack of information is problematic if we want to generate a natural language string (from underspecified semantic representations) in a machine translation system like Verbmobil, a semantic-based machine translation system which translates spoken German and Japanese into English [Wah93, KGN94, BGL$^+$96, GLM96]. Clear scope configurations (preferences) in the source language are easily lost in the target language. Consider the following examples:

[1]Sometimes a distinction is made between the Underspecified Logics underlying the work by most of the authors mentioned above and the Quasi Logics of e.g. [AC92]. In contrast to the "underspecified" representations, the Quasi Logical Form is assumed to have a well-defined model-theoretic mapping on its own (i.e., separate from the mapping over the fully resolved logical form). Making such a distinction is not relevant to the discussion in the present paper. The ideas we describe should be equally applicable to both these types of semantics — regardless of the underlying models.

(1) *das paßt auch nicht*
that fits also not
'that does not fit either'

(2) *ich kann$_i$ sie nicht verstehen* ϵ_i
I can you not understand
'I can not understand you'

In (1) the negation adverb *nicht* outscopes the focus particle *auch*. This information is required to get a correct translation. The preferred reading in (2) is the one where *nicht* has scope over the modal verb *kann* (*können*). In these examples the syntactic configurational information (relative c-command relations) for German supports the preferred scoping: the operator with the widest scope is c-commanding the operator with narrow scope. However, this is not easily carried over to English, since there is no verb movement in the English sentence of (2), so *not* does not c-command *can* in this case.

In this paper we focus on the underspecification of scope introduced by quantifying noun phrases, adverbs, and particles. The underspecified representations we use, VITs, resemble the ideas of Underspecified Discourse Representation Structures [Rey93] and Hole Semantics [Bos95]. Firstly, VITs, the Verbmobil Interface Terms are described in Section 2. Section 3 then shows how VITs are built up in the compositional semantics of the Verbmobil system. Section 4 discusses how structural constraints can help to resolve the underspecified VIT structures. The section following gives an evaluation of how well the resolution algorithm fares on real dialogues examples. Finally, Section 6 sums up the previous discussion.

2 Underspecified Semantic Representations

The Verbmobil Interface Term, VIT for short [BES96] is a representation that encodes the linguistic information of a natural language utterance. The goal is to describe a consistent interface structure in order to perform communication between the different language analysis modules within the Verbmobil system in a smooth and consistent way. Besides semantic structure, the VIT contains prosodic, syntactic, and discourse information. In this paper we will pay attention only to the semantic parts of the VIT. These are

1. the top label,

2. a set of labeled conditions, and

3. a set of constraints.

The set of labeled conditions represent (among others) ordinary predicates, quantifiers, pronouns, and operators. All conditions are uniquely labeled. Scope (appearing in quantifiers and operators) is represented in an underspecified way by variables ranging over labels. These variables are referred to as *holes*. The labeling of conditions is used to make it easier to refer to a particular condition, enabling us to state constraints on the relations between the conditions at the meta-level. A key constraint is the *subordination relation*, `leq(L,H)`, which expresses that a condition with label `L` is within the scope of the element introducing the hole `H`.

An example VIT is shown in (3), paraphrasing utterance (1) *das paßt auch nicht*. For the purpose of this paper, VITs are written as Prolog terms with arity three, that is, with one argument each for the three semantic parts of the VIT as given above. Labels are atoms starting with a lowercase `l` followed by a number, holes start with `h`, and variables over individuals are atoms starting with lowercase `i`. Of course, sets are encoded as Prolog-lists.

(3)
```
vit([l1],                    % top label
    [decl(l1,h1),
     pron(l2,i1),
     passen(l3,i2,i1),       % labeled conditions
     auch(l4,h2),
     nicht(l5,h3),
     group(l6,[l2,l3])],
    [leq(l6,h1),
     leq(l6,h2),
     leq(l6,h3),             % constraints
     leq(l4,h1),
     leq(l5,h1)]).
```

The top label `l1` points to the sentence mood operator (`decl`, for the declarative mood in this case), which is trivially outscoping all other elements in its domain.

The pronoun *das* is represented in a VIT by a predicate `pron`, marking unresolved anaphoric information. The predicate introduced by the main verb of (1), `passen`, and its subject, the pronoun condition, are in the same scope unit `l6`, represented by a grouping predicate. Finally, the particles *auch* and *nicht* are treated as operators.

The four `leq` constraints state that the verb information is in the relative scopes (`h2` and `h3`) of the two particles. And furthermore, neither the verb, nor the two particles should outscope the sentence mood operator `h1`. Note that no restrictions are placed on the relative scopes of the particles.

VITs are interpreted with respect to a so-called *plugging*. A plugging is a mapping from holes to labels [Bos95]. The number of readings that a VIT encodes equals the number of possible pluggings for this VIT. In our example there are no restrictions on the relative scopes of *auch* and *nicht*. Thus, there are two pluggings which do not violate the subordination constraints.

(4) $\{\mathtt{h1} = \mathtt{l4},\ \ \mathtt{h2} = \mathtt{l5},\ \ \mathtt{h3} = \mathtt{l6}\}$

(5) $\{\mathtt{h1} = \mathtt{l5},\ \ \mathtt{h2} = \mathtt{l6},\ \ \mathtt{h3} = \mathtt{l4}\}$

The plugging in (4) resembles the reading where *auch* outscopes *nicht*. In (5) we have the reading where the negation has wide scope. In combination with a plugging, a VIT can be translated to a Discourse Representation Structure (DRS).[2] For example, the `pron` condition introduces a discourse marker (`i1`) which should be linked to a proper antecedent, the `group` condition is the same as a *merge* between DRSs, `passen` is a basic one place predicate, `nicht` is translated as negation, etc.

3 Construction of underspecified representations

In addition to underspecification, two other basic principles guide the semantic construction in Verbmobil: keep as much as possible of the semantic information lexicalized and pass the information up from the terminal nodes of a parse tree to the input nodes in a compositional manner [BGL+96].

Keeping most of the information in the lexicon (rather than in the grammar rules, as traditionally) reflects a strong trend both in unification-based grammar approaches in general as well as in most approaches to computational semantics.

[2] DRSs are the structures used in DRT, Discourse Representation Theory [KR93].

The overall idea is to keep the grammar rules as simple as possible — which in turn may result in rather complicated lexica. The result here is that a large part of the interesting work actually is done at the lexical level.

The principle of *compositionality* is quite central to the formalization of the semantic construction. Compositionality means that the interpretation of a phrase is a function of the interpretations of its subphrases. In the grammar rules, we then allow for information passing in three ways: trivial composition, function-argument application, and modifier-argument application.

The trivial composition manifests itself mainly in rules which are inherently (semantically) unary branching. That is, rules which either are syntactically unary branching, or where the semantics of at the most one of the daughter (right-hand side) nodes need to influence the interpretation of the mother (left-hand side) node.

The two types of application rules (function- resp. modifier-argument) are in fact quite similar to each other and appear on all (semantically) binary branching rules of the grammar. (Of course, the grammar formalism allows for rules which are not in normal form, i.e., which are more than binary branching. However, these may be reduced to normal form on the semantic side by multi-application rules.)

In functor-argument application the main part of the semantic information is passed between the mother node and the functor (semantic head). In modifier-argument application the main part is passed up from the argument. The main difference between these two rule types pertains to the (semantic) subcategorization schemes: In functor-argument application, the functor subcategorizes for the argument, the argument may optionally subcategorize for the functor, and the mother subcategorizes for whatever is left on the functor's subcategorization list after the argument having been removed from it.

In modifier-argument application, in contrast, the modifier must subcategorize for the argument (only), while the argument does not subcategorize for the modifier, so that the mother's subcategorization list is identical to that of the argument.

4 A Resolution Algorithm

Previous approaches to scopal resolution have mainly been treating the scopal constraints separately from the rest of the semantic structure (as in Cooper storage techniques [Coo83]). Mostly this work has been very theoretical, or it is argued that the context must be taken into account to correctly resolve the scopal relations.

The work on scopal resolution in the SRI Core Language Engine [Mor88, MP92] however, looked more into the compositional semantic process as a whole. The algorithm used by Moran and Pereira asserted variables for the unresolved scoped already at the lexical level together with some constraints on the resolution. Other constraints were sometimes added in the grammar rules, albeit in a seemingly *ad hoc* manner. Most of the constraints used in the scopal resolution of the Core Language Engine were anyhow provided by a handcoded knowledge-base specifying the inter-relation of different scope-bearing operators. The facts from this knowledge-base were applied in a process separate from the construction of the compositional semantics.

In contrast, we want to be able to capture the constraints already given by the function-argument structure of an utterance and provide a possible resolution of the scopal ambiguities built up already while constructing the semantic representation. Thus we introduce a set of three features (which we jointly will refer to as `holeinfo`) on each category in the grammar. On the terminals, these will normally have the values

```
sb_label = MainLabel
hole = none
hole_label = no_hole
```

Indicating that the category does not contain a hole. More specifically, the `sb_label` is the label of the element of the substructure below it having widest scope; `hole` indicates which type of hole a structure contains. The values are `none`, `normal`, or `island`, with the latter being necessary for categories which should force the scopal elements below them to be resolved locally. An example of islands in this sense are sentence coordinators, as will be discussed further on.

Scope bearing categories (quantifiers, particles, etc.) introduce (`normal`) holes and get the following feature setting with `hole_label` pointing to the hole which has been introduced.

```
sb_label = MainLabel
hole = normal
hole_label = Hole
```

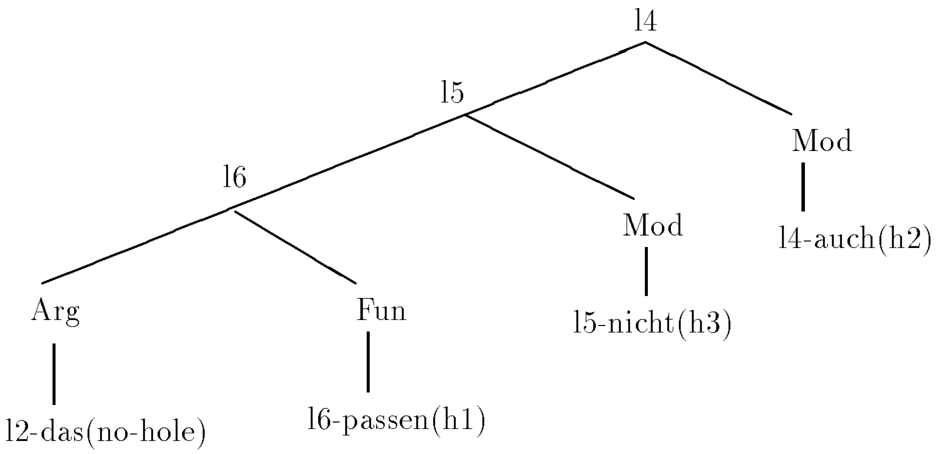
When the `holeinfo` information is built up in the tree, the `sb_label`s are normally passed up as the main labels (that is, from the semantic head daughter to the mother node), unless the non-head daughter of a binary branching node contains a hole. In that case, the hole is plugged with the `sb_label` of the head daughter and the `sb_label` of the mother node is that of the non-head daughter. On the top-most level of the grammar, the top hole itself is plugged with the `sb_label` of the full structure. To be more concrete, the two application type rules pass the `holeinfo` as follows:

```
mod_arg(Modifier,Argument,Result) =>
   head_nonhead(Argument,Modifier,Result).

fun_arg(Functor,Argument,Result) =>
   head_nonhead(Functor,Argument,Result).

head_nonhead(Head,Nonhead,Result) =>
   IF
     Nonhead:holeinfo:hole=none
   THEN
     Result:holeinfo=Head:holeinfo
   ELSEIF
     Nonhead:holeinfo:hole=normal
   THEN
     sb_plug(Nonhead:holeinfo:hole_label,Head:holeinfo:sb_label),
     Result:holeinfo:sb_label=Nonhead:holeinfo:sb_label,
     Result:holeinfo:hole=Head:holeinfo:hole,
     Result:holeinfo:hole_label=Head:holeinfo:hole_label.
   ENDIF
```

We will illustrate the rules with an example. In utterance (1) *das paßt auch nicht* we have the following semantic argument structure:



The verb *passen* first get applied to its subject *das*. Since the pronoun contains no hole (i.e., it is non-scope bearing), we truehave the first case of the `head_nonhead` rule in which the result's holeinfo is identical to that of the head daughter.[3] When the modifier *nicht* gets applied to the verbal structure, we have the case with the non-head daughter containing a hole. For this, `h3`, we introduce an `sb_plug` (semantic-head based plugging) into the VIT. The label plugging the hole is the `sb_label` of the head daughter, `l6`. The `sb_label` of the resulting structure is `l5`, the `sb_label` of the modifier.

For *auch*, the process is repeated so that its hole (`h2`) is plugged with `l5`. At this stage we have reached the top of the structure and the remaining hole of the entire structure (`h1`) is plugged by the structure's `sb_label`, which is now `l5`. In total, the following three constraints are added to the VIT in (3):

```
sb_plug(h1,l4)
sb_plug(h2,l5)
sb_plug(h3,l6)
```

And this scope preference corresponds to (4), the reading where *auch* has scope over *nicht*, resulting in the correct reading.

Coordinations, discourse relation adverbs, and the like add a special case. They introduce a new top hole which should be above the top holes of both sentential complements and get

```
sb_label = MainLabel
hole = island
hole_label = Hole
```

This is used to override the syntactic structure and to always produce a plugging where the top holes of the sentential complements have been plugged with their own `sb_label`s. The island cases thus complicate the implementation of the rules above a bit in that they always must accommodate the fact that one of the daugthers may carry an 'island' type hole.

[3]Note firstly that the main label of the verb is the grouping label `l6`, rather than the label `l3` corresponding directly to the predicate `passen` itself, and secondly that the main verb of the utterance `passen` introduces the hole `h1` for the scope of the sentence mood operator. These general facts are stated in the verb's lexical entry.

5 Evaluation

In order to evaluate the algorithm, the results of the pluggings obtained for four dialogues in the Verbmobil test set were checked. The results can been seen in Table 1. We only consider utterances for which the VITs contain more than two holes, since those are the only ones in which ambiguities appear: A VIT with one hole only trivially contains the top hole of the utterance (that is, the hole for the sentence mood predicate, which is introduced by the main verb); a VIT with two holes contains the top hole and the hole for one more scope-taking element. Since the mood-predicate always will have scope over the remaining proposition, resolution is still trivial. Thus neither one-hole, nor two-hole VITs contain any real ambiguities to resolve.

For VITs with three or more holes, we have true ambiguities: The number of scope bearing operators is the number of holes minus one. In Table 1, the first column gives the number of utterances with no ambiguity ($\leq$ 2 holes), the following columns indicate the number of holes for the ambiguous sentences. Most commonly the utterances contained one true ambiguity (= 3 holes). Utterances with more than two ambiguities ($\geq$ 5 holes) are rare and have thus been grouped together in one column in the table.

The dialogues evaluated are identified as three of the so called "Blaubeuren" dialogues (B1, B2, and B7) and one of the "Reithinger-Herweg-Quantz" dialogues (RHQ1). These four together form the standard test-set for the German language modules of the Verbmobil system.

Table 1: Results of evaluation

Dialogue	Utterances	Number of holes				Correctness	
		≤ 2	3	4	≥ 5		
B1	48	34	9/11	1/2	1/1	11/14	78%
B2	41	26	5/8	2/3	4/4	11/15	73%
B7	48	36	7/8	0/1	3/3	10/12	83%
RHQ1	91	68	10/11	5/6	4/6	19/23	82%
Total	228	164	31/38	8/12	12/14	51/64	79%

Each field in the table shows the number of correctly resolved ambiguous utterances over the total number of ambiguous utterances in the test set. In the column with '$\leq$ 2' holes, no ambiguities exist, so only the total number of utterances of this type are given.

As can be seen in the table, the resolution based on the semantic argument structure alone fares surprisingly well. The default scoping introduced by the algorithm of the previous section is also the preferred one for about 80% of the ambiguous utterances, leaving errors in only 5.7% (13/228) of the overall utterances. Looking closer at the thirteen cases where the algorithm fails, we see that the reasons for the failures divide as:

- technical construction problem in VIT (four times)
- an indefinite noun phrase should have had wide scope (three times) or narrow scope (once)
- a scopal adverb should have had widest scope (three times)
- combination of (a modal) verb movement and negated question resulted in wrong scope preference (once)
- relative scope of particles is not conform to c-command structure assigned by the syntactic analysis (once)

Our results indicate that for a practical system, more sophisticated approaches to scopal resolution (i.e., based on the relations between different scope-bearing elements and/or contextual information) will not add much to the overall system performance.

6 Conclusions

In this paper we presented an algorithm for scope resolution in underspecified semantic representations. Scope preferences are suggested on the basis of semantic argument structure. The major novelty of this approach is that, on the one side, we maintain an (with respect to scopal ambiguities) underspecified semantic representation, and on the other side, at the same time suggest a scope resolution possibility. The algorithm was evaluated on four "real-life" dialogues and fared suprisingly well: about 80% of the utterances containing scopal ambiguities were correctly interpreted by the suggested resolution, leaving scopal resolution errors in only 5.7% of the overall utterances.

References

[AC92] Hiyan Alshawi and Richard Crouch. Monotonic semantic interpretation. In *Proceedings of the 30th Annual Meeting of the Association* *for Computational Linguistics*, pages 32–39, Newark, Delaware, June 1992. ACL. Also available as SRI International Technical Report CRC-022, Cambridge, England.

[ACGR91] Hiyan Alshawi, David M. Carter, Björn Gambäck, and Manny Rayner. Translation by quasi logical form transfer. pages 161–168, 1991. ACL 91; Also available as SRI International Technical Report CRC-021, Cambridge, England.

[BES96] Johan Bos, Markus Egg, and Michael Schiehlen. Definition of the abstract semantic classes for the Verbmobil Forschungsprototyp 1.0. VM Report 165, Universität des Saarlandes, Saarbrücken, Germany, August 1996.

[BGL+96] Johan Bos, Björn Gambäck, Christian Lieske, Yoshiki Mori, Manfred Pinkal, and Karsten Worm. Compositional semantics in Verbmobil. In *Proceedings of the 16th International Conference on Computational Linguistics*, volume 1, pages 131–136, København, Denmark, August 1996. ACL.

[Bos95] Johan Bos. Predicate logic unplugged. In P. Dekker and M. Stokhof, editors, *Proceedings of the 10th Amsterdam Colloquium*, volume 1, pages 133–142, University of Amsterdam, Amsterdam, Holland, December 1995. Also available as Verbmobil Report 103, University of Saarbrücken, Germany.

[CCvE+96] Robin Cooper, Dick Crouch, Jan van Eijck, Chris Fox, Josef van Genabith, Jan Jaspars, Hans Kamp, David Milward, Manfred Pinkal, Massimo Poesio, Steve Pulman, Ted Briscoe, Holger Maier, and Karsten Konrad. Using the framework. FraCaS Deliverable D16, University of Edinburgh, Edinburgh, Scotland, January 1996.

[Coo83] Robin Cooper, editor. *Quantification and Syntactic Theory.* D. Reidel, Dordrecht, Holland, 1983.

[GLM96] Björn Gambäck, Christian Lieske, and Yoshiki Mori. Underspecified Japanese semantics in a machine translation system. In *Proceedings of the 11th Pacific Asia Conference on Language, Information and Computation*, pages 53–62, Seoul, Korea, December 1996.

[KGN94] Martin Kay, Jean Mark Gawron, and Peter Norvig. *Verbmobil: A Translation System for Face-to-Face Dialog.* Number 33 in Lecture Notes. CSLI, Stanford, California, 1994.

[KR93] Hans Kamp and Uwe Reyle. *From Discourse to Logic: An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and DRT.* Kluwer, Dordrecht, Holland, 1993.

[Mor88] Douglas B. Moran. Quantifier scoping in the SRI Core Language Engine. In *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics*, pages 33–40, State University of New York, Buffalo, New York, June 1988. ACL.

[MP92] Douglas B. Moran and Fernando C.N. Pereira. Quantifier scoping. In *The Core Language Engine.* The MIT Press, Cambridge, Massachusetts, March 1992.

[Pin95] Manfred Pinkal. Radical underspecification. In P. Dekker and M. Stokhof, editors, *Proceedings of the 10th Amsterdam Colloquium*, volume 3, pages 587–606, University of Amsterdam, Amsterdam, Holland, December 1995.

[Poe95] Massimo Poesio. Disambiguation as (defeasible) reasoning about underspecified representations. In P. Dekker and M. Stokhof, editors, *Proceedings of the 10th Amsterdam Colloquium*, volume 3, pages 607–625, University of Amsterdam, Amsterdam, Holland, December 1995.

[Rey93] Uwe Reyle. Dealing with ambiguities by underspecification: Construction, representation and deduction. *Journal of Semantics*, 10:123–179, 1993.

[Wah93] Wolfgang Wahlster. Verbmobil: Translation of face-to-face dialogs. In *Proceedings of the 3rd European Conference on Speech Communication and Technology*, pages 29–38, Berlin, Germany, September 1993.