Scoring for Overlay based on Informational Distance[*]

Jan Alexandersson and **Tilman Becker** and **Norbert Pfleger**
DFKI GmbH, Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany
`{janal,becker,pfleger}@dfki.de`

Abstract

Understanding a new utterance in a discourse is a process of combining the previous context with possible interpretations of the new utterance. In this process, parts of the previous context will be extended, refined, changed, or deleted. One formalization of this process, overlay, is based on default unification and since it always generates a result, a scoring function is essential to judge the quality of the result. This paper refines the current scoring function of overlay to take the *informational distance* between old and new information into account.

1 Introduction

For dialogue systems, new information provided by the user is often incomplete with respect to the task or inconsistent with respect to the context or even both. Furthermore, analysis and interpretation has to face alternative hypotheses which have to be judged. In the multimodal dialogue system SmartKom (Wahlster, 2003), many components involved in the analysis contribute to the judgment of the hypotheses by scoring them according to their knowledge. In the case of discourse modeling we score a hypothesis by comparing it with the current discourse state.

Fundamental to the work presented here is the existence of an ontology modeling the objects and actions the system can be engaged in. Formally, we treat the ontology as a type hierarchy and instances of the ontology as typed feature structures (TFS). For the interpretation of user contributions inconsistent with the prior discourse, we have successfully used a default reasoning technique called "default unification" (Bouma, 1992; Alexandersson and Becker, 2003) which is also known as "priority union" (Grover et al., 1994). Enhanced with a scoring function (Pfleger et al., 2002), our default unification algorithm—overlay—is able to not only enrich incomplete user contributions with coherent contextual information from the previous discourse, it can also rank competing interpretations according to their cohesion with the previous discourse state.

A Refined Score for Overlay

Our present definition of scoring (Pfleger et al., 2002) throws away information in that all type clashes are treated as equivalent and counted as a penalty of 1. In this work, we provide a refinement of the scoring function that is sensitive to the severeness of the type clash. Our refinement takes into account the relative amount of information carried over to the resulting type.

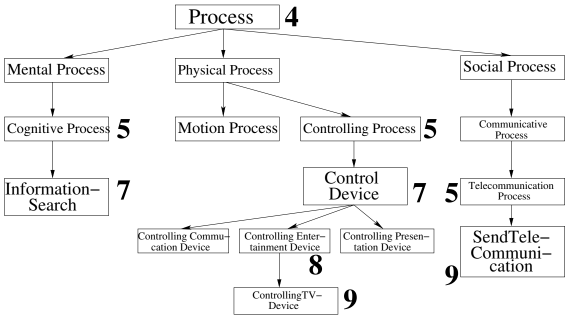

Figure 1: Excerpt from our ontology together with the number of roles for each concept.

Throughout the paper we will make use of the following sample discourse:

(1) **U:** I'd like to go to the movies.
(2) **S:** When do you want to go?
(3) a. **U:** I'd like to watch TV.
  b. **U:** I'd like to send a message.
  c. **U:** Channel 7.
    i. What is showing on channel 7?
    ii. Switch to channel 7!

Given our previous approach, (3a) and (3b) could, dependent on the details of the ontology, receive the same score since it is only affected by exactly one type clash resulting in a penalty of 1 (see section 2).

---

[*] The research presented here is funded by the German BMBF under grants 01 IMB 01A (VirtualHuman) and 01 IL 905 (SmartKom) and by the EU under the grants FP6-506811 (AMI) and IST-2001-32311 (Comic). We would like to thank Iryna Gurevych for help with figure 1.

This is not desirable, since watching TV is more related to going to the movies than sending a message and this should be reflected in the score. It would be possible and indeed desirable to infer that (3c) is a request to ask for the program for channel 7. There are a lot of additional interpretations possible and another one—which should be less probable—is "switch to channel 7." Our solution for finding such interpretations is described in section 3.

2 Preliminaries

Our work is based on the existence of an ontology, e. g., (Gurevych et al., 2003) which represents the knowledge about the world and what actions can be undertaken by the user and/or the system. Most classes have roles which can have as their values (instances of) other classes or atomic values, i. e., strings. An excerpt of such an ontology is depicted in figure 1. We assume that each pair of classes has a unique LUB, i. e., the ontology allows for unary inheritance only.[1]

Furthermore, we view instances of the ontology as *typed feature structures* (henceforth TFS). This is important since TFS are well-studied. In particular we are interested in manipulation operations like unification (Carpenter, 1992) and related operations. Taking this viewpoint, the ontology corresponds to a type hierarchy, the roles to features $(f, g, \ldots)$ and values $(v, w, \ldots)$ are either instances of other TFSs or atomic values.

Our previous work provides a more thorough description of what comes below (Alexandersson and Becker, 2001; Alexandersson and Becker, 2003; Pfleger et al., 2002) so next we summarize the framework around TFS, unification and overlay (for a complete description of the logic of TFSs see, e. g., (Carpenter, 1992; Krieger, 1995)).

Via a relation, $\preceq$, over a set of types Type our type hierarchy is a *bounded complete partial order* (BCPO), i. e., it is reflexive, antisymmetric and transitive. In a BCPO, unification of two TFSs, $\sqcup$, is defined as their GLB. In general, credulous default unification between two TFSs, $T \overset{<}{\sqcup} F$, is defined as a set of TFS characterized by $F \overset{<}{\sqcup} = \{F \sqcup G' | G' \sqsubseteq G$ is maximal such that $F \sqcup G'$ is defined $\}$, e. g., (Carpenter, 1993). For our restricted hierarchy, the resulting set will contain exactly one TFS, i. e., the result is unambiguous. In (Alexandersson and Becker, 2003) we provided the following operational description for the computation of default unification. The definition of assimilation is slightly different than in (Pfleger et al., 2002) in that this definition takes into account that background and covering could be positioned anywhere in the type hierarchy. Prior definitions assumed the covering to be always incompatible or more special.

Definition 1 (Assimilation) *Let*

- *$c, b \in \mathcal{S}$ such that the covering $c = \langle t_c, \{f_1 : v_1, \ldots, f_n : v_n\}\rangle$ and the background $b = \langle t_b, \{g_1 : w_1, \ldots, g_n : w_m\}\rangle$*

then, the assimilation of c and b, $\alpha(c, b)$, is defined as:

$$\text{If } t_c \preceq t_b\text{: } \alpha(c,b) := (\langle t_b, \{f_1 : v_1, \ldots, f_n : v_n\}\rangle, b) \quad (1)$$

$$\text{otherwise: with } g_i = LUB(t_c, t_b)$$
$$\alpha(c,b) := (c, \langle t_c, \{g_i : w_i, \ldots, g_j : w_j\}\rangle) \quad (2)$$

In what follows, we will assume two operations – α_c and α_b – that select the assimilated covering and background respectively. We continue with the definition of overlay for typed feature structures:

Definition 2 (Overlay) *Let*

- *c and b be two TFS such that the covering $c = \langle t_c, \{c_1 : f_1, \ldots, c_n : f_n\}\rangle$ and the background $b = \langle t_b, \{b_1 : g_1, \ldots, b_m : g_m\}\rangle$ and $A = \alpha(c, b)$*

then $\underline{\textsc{Overlay}(c, b)}$ is defined as:

$$\textsc{Overlay}(c,b) := \textsc{Overlay}'(\alpha_c(A), \alpha_b(A)) \quad (3)$$
$$\textsc{Overlay}'(c,b) := \langle t_{\alpha_c}, \{o_i : h_i \mid$$
$$o_i = c_j = b_k, h_i = \textsc{Overlay}(f_j, g_k), f_j, g_k \in \mathcal{S}, \text{ or} \quad (4)$$
$$o_i = c_j = b_k, h_i = f_j, \text{ where } f_j, g_k \in \mathcal{A}, \text{ or} \quad (5)$$
$$o_i = c_j, h_i = f_j, o_i \neq b_k, 1 \leq k \leq m, \text{ or} \quad (6)$$
$$o_i = b_k, h_i = g_k, o_i \neq c_j, 1 \leq j \leq n \quad \}\rangle \quad (7)$$

The first case (4) is the recursive step used when the values are typed feature structures. In the second case (5), when the values of covering and background are atomic, the value in the covering is used. The next case (6) applies when the feature is absent in the background and we use the value from the covering. Finally, (7) is the case when the feature of the covering has no value: then the value in the background is used.

Given this description we recapitulate the way of counting parameters given in (Pfleger et al., 2002):

co a TFS or an atomic value stemming from the covering is added to the result. **co** is incremented for each feature in the covering

bg a TFS or an atomic value in the result occurs in the background; **bg** is incremented

tc type clash, i.e., the type of the covering and background was not identical. This is identified during the computation of the assimilation.

cv conflicting values. This occurs when the value of a feature from the background is overwritten.

---

[1] This restriction is not mandatory or even important, but our practical work is based on such a hierarchy. In other work, we have generalized our reasoning to include hierarchies with multiple inheritance (Alexandersson and Becker, 2004).

All these parameters are collected during the application of Overlay and in the end a scoring function computes a single number that reflects the structural consistency of the two structures. The sum of **co** and **bg** minus the sum of **tc** and **cv** will be weighted against the sum of **co**, **bg**, **tc** and **cv**. This leads to a function (shown in formula below) whose codomain is $[-1, 1]$.

Definition 3 (Score)

$$score(co, bg, tc, cv) = \frac{co + bg - (tc + cv)}{co + bg + (tc + cv)}$$

The fundamental property of this function is that its positive extremal ($score(co, gb, tc, cv) = 1$) indicates that the feature structures are unifiable. The negative extremal ($score(co, gb, tc, cv) = -1$) indicates that all information from the background was overwritten by information from the cover. Scores within this interval indicate that the cover more or less fits the background: the higher the score the better the cover fits the background. Negative values signal that conflicting and thus overlayed values outweigh unifiable values (positive values vice versa).

3 Informational Score

As defined above, all type clashes cause a constant penalty, thus throwing away information that could be used to assess a more precise score. While there are many ways to measure the severeness of the type clash, e. g., the geometric distance between the types, we have found the *informational distance*, as defined below to best match our requirements: We compute the ratio of lost and kept features of the background.

Definition 4 (Informational Distance) Let

- bg be the type of the backgound of the type clash
- lub be the type of the least upper bound of the clashed cover and background
- $|lub|$ be the number of features defined for the LUB type and $|bg|$ be the number of features defined for the background type

Then, the *informational distance*, *idist* is defined as:

$$idist(lub, bg) = \begin{cases} 0 & if\, |bg| = 0 \\ \frac{|bg| - |lub|}{|bg|} & otherwise \end{cases}$$

3.1 The Revised Scoring Function

We continue by showing how the informational distance can be used to extend the existing scoring function so that every TFS embedded under the type clash is assessed by the relative distance between the two types of the type clash.

The revised scoring function is still based on a concise number of parameters that can be collected during the application of overlay. The only difference to our present one is that a new parameter—*weighted type clash*—is introduced.

Definition 5 (Weighted Type Clash) Let

- co be number of TFSs or atomic features stemming from the covering,
- bg be the number of feature values or atomic features stemming from the background,
- $i, \ldots, n$ are type clashes, and cv be the number of conflicting atomic values

Then, the *weighted type clash*, *wtc* is defined as

$$wtc = \sum_{i=1}^{n} idist(lub_i, bg_i)$$

Finally, we replace the previous type clash count with the weighted type clash number and obtain the new scoring function:

Definition 6 (Informational Score)

$$iscore(co, wtc, bg, cv) = \frac{co + bg - (wtc + cv)}{co + bg + wtc + cv}$$

Our new scoring function *informational score* is similar to the initial one, i. e., again, all parameters are collected during the application of Overlay. Only the parameter wtc needs a different handling and necessitates a slight change to the overlay algorithm. In case of a type clash the result of applying the idist function is added to the wtc parameter.

Even the basic properties of the scoring function does not change much. The codomain of the function is still the interval between $[-1, 1]$ and the positive extremal ($score(co, gb, tc, cv) = 1$) still states that the operants are unifiable. Knowing that a covering and a background are unifiable is a valuable information. This is also a reason why we only take the distance between the background and the LUB into account. Considering also the informational distance between the covering and the LUB would lead to a lower score if the background and the covering are in a direct subtype relation even though they are still unifiable.

Here, we make a note on how we obtain the interpretations (3c-i) and (3c-ii). For SmartKom, we characterize the ontology in terms of *application objects* and *subobjects.* The former can be viewed as “top-level” objects corresponding to some compound action, e. g., InformationSearch or ControlDevice which the user can execute. Subobjects are parts of the application objects and can (recursively) contain other subobjects. In previous work (Löckelt et al., 2002) we showed how to interpret short utterances directly related to the previous context. Our interpretation relied on *predictions* from the action planner which contain expectations of the next user contribution given the current task (i. e., application object). However, our approach failed to interpret short user contributions more distantly related, i. e., utterances that cannot be attached to the current application object. An example for this is TV-program in a movie reservation application.

In case the predictions from the action planner fail to suggest an interpretation for the short utterance, we compute the set of all application objects that the short utterance can be attached to. In the case of (3c) there are not only the interpretations (3c-i) and (3c-ii) but also interpretations, such as “switch on the TV on channel 7.” The path from the short utterance to the application object is used to construe, in this case the interpretations (3c-i) and(3c-ii).

To underline the effect of the new scoring function we pick up again the dialogue excerpt mentioned in the introduction. Utterance (1) forms the discourse background to interpret the three potential subsequent utterances (3a), (3b) and (3c). Table 1 compares the scores for the old and the new functions for each covering–background combination from our example. The score for (3a) has received a higher value indicating a closer relationship than before. Also, (3b) receives a higher score because the types SendTeleCommunication and InformationSearch share four features of their common supertype Process. Finally, the relative difference between (3c-i) and (3c-ii) has become greater which is more natural: in the context of going-to-the-movies, it is more probable that the user would like to know about the TV-program rather than switching the channel.

utterance	*score*	*iscore*
U 3a	0.333	0.846
U 3b	0	0.286
U 3c-i	0.556	0.798
U 3c-ii	0.5	0.687

Table 1: Comparison of old (*score*) and new (*iscore*) scoring function.

References

Jan Alexandersson and Tilman Becker. 2001. Overlay as the Basic Operation for Discourse Processing in a Multimodal Dialogue System. In *Workshop Notes of the IJCAI-01 Workshop on “Knowledge and Reasoning in Practical Dialogue Systems”*, Seattle, Washington, August.

Jan Alexandersson and Tilman Becker. 2003. The Formal Foundations Underlying Overlay. In *Proceedings of the Fifth International Workshop on Computational Semantics (IWCS-5)*, Tilburg, The Netherlands, February.

Jan Alexandersson and Tilman Becker. 2004. Default Unification for Discourse Modelling. In Harry Bunt and Reinhard Muskens, editors, *Computing Meaning*, volume 3. Kluwer Academic Publishers, Dordrecht. Forthcoming.

Gosse Bouma. 1992. Feature structures and nonmonotonicity. *Computational Linguistics*, 18(2):183–203.

Bob Carpenter. 1992. *The logic of typed feature structures.* Cambridge University Press, Cambridge, England.

Bob Carpenter. 1993. Skeptical and credulous default unification with application to templates and inheritance. In A. Copestake E. J. Briscoe and V. de Paiva, editors, *Inheritance, Defaults and the Lexicon*, pages 13–37. Cambridge University Press, Cambridge, England.

Claire Grover, Chris Brew, Suresh Manandhar, and Marc Moens. 1994. Priority union and generalization in discourse grammars. In *32nd. Annual Meeting of the Association for Computational Linguistics*, pages 17–24, Las Cruces, NM. Association for Computational Linguistics.

Iryna Gurevych, Robert Porzel, Elena Slinko, Norbert Pfleger, Jan Alexandersson, and Stefan Merten. 2003. Less is more: Using a single knowledge representation in dialogue systems. In *Proceedings of the HLT-NAACL Workshop on Text Meaning*, pages 14–21, Edmonton, Canada, May.

Hans-Ulrich Krieger. 1995. *TDL—A Type Description Language for Constraint-Based Grammars. Foundations, Implementation, and Applications.* Ph.D. thesis, Universität des Saarlandes, Department of Computer Science, september.

Markus Löckelt, Tilman Becker, Norbert Pfleger, and Jan Alexandersson. 2002. Making sense of partial. In *Proceedings of the sixth workshop on the semantics and pragmatics of dialogue (EDILOG 2002)*, pages 101–107, Edinburgh, UK, September.

Norbert Pfleger, Jan Alexandersson, and Tilman Becker. 2002. Scoring functions for overlay and their application in discourse processing. In *KONVENS-02*, Saarbrücken, September – October.

Wolfgang Wahlster. 2003. Smartkom: Symmetric multimodality in an adaptive and reusable dialogue shell. In R. Krahl and D. Gnther, editors, *Proceedings of the Human Computer Interaction Status Conference 2003*, pages 47–62, Berlin: DLR, June.