A PREFERENCE MECHANISM

BASED ON MULTIPLE CRITERIA RESOLUTION

Yannis Dologlou
Eurotra-GR, Margari 22
11525 Athens, Greece.

Giovanni Malnati
Eurotra-IT, Gruppo Dima,
Corso F.Turati 11/C
10128 Torino, Italy.

Patrizia Paggio
patrizia@eurotra.uucp
Eurotra-DK, University of Copenhagen,
Njalsgade 80
2300 Kbh S, Denmark.

ABSTRACT

This paper presents an experimental preference tool designed, implemented and tested in the Eurotra project. The mechanism is based on preference rules which can either compare subtrees pairwise or single out a subtree on the basis of some specified constraints. Scoring permits combining the effects of various preference rules.

THE PROBLEM

The aim of a translation system is to produce the correct translation of a given text. In Eurotra, where translation is split up into a series of mappings among intermediate levels of representation, provisional overgeneration is a necessary evil [Raw et al. 1989]: the closer to surface structure a level of representation is, the harder it becomes for the parser to produce an unambiguous result. In the Eurotra framework, the E-framework [Bech et al. 1989], overgeneration can be partially controlled by filters which describe parse trees that are to be discarded as not obeying some specified constraints. Thus, filters apply to individual objects and are meant to delete inherently wrong representations. But there are cases where the grammar produces multiple analyses of a given input because the input is ambiguous with respect to a given level. All of these analyses are in some sense correct, although further processing might discard some of them. Our aim was to design a preference mechanism able to choose the best among a set of acceptable candidates.

OUR VIEW OF PREFERENCE

Preference has been defined in a number of ways, e.g. as a gradual fulfilment of semantic constraints [Fass and Wilks 1983], as a lexically induced syntactic bias [Ford et al. 1982], as a parsing strategy independent of linguistic criteria [Frazier and Fodor 1978, Pereira 1985], and as a system based on multiple judgements reflecting the complexity of psychological processes [Jackendoff 1985].

Our approach, which is greatly indebted to Jackendoff's theory of preference rule systems, is based on the following assumptions:

- Preference is a method which, on the basis of some preference criteria, chooses the best one among a set of possible interpretations which are all correct according to the grammar.
- Each preference criterion is expressed as a set of statements, where a statement is either a binary relation between competing interpretations or the description of a subtree which satisfies some defined criteria.
- There is no unique preference criterion according to which the best interpretation can be chosen: preference criteria are multiple, and possibly contradictory. A preference mechanism must be able to accommodate such multiplicity.
- Preference criteria are heuristic principles which may vary according to the language and the text type: therefore, they are not hardwired in the system.

In the previous Eurotra preference mechanism [Petitpierre et al. 1987], preference statements were only defined as binary relations between subtrees. Since comparing subtrees is a rather expensive operation from the computational point of view, and since a number of preference criteria - e.g. the principle of right-low attachment - cannot be expressed in binary terms, we have allowed both binary and non-binary preference rules. The application algorithm of p-rules and the way in which various preference criteria are combined are also new with respect to the previous system.

THE MECHANISM PROPOSED

The mechanism proposed is an independent module which is activated on the results output by the parser. The module consists of preference rules of two possible kinds, which we call binary and unary rules.

A binary rule establishes a preference relation between two corresponding (sub)trees (from here on, (sub)tree will be used in the sense of a representation of an interpretation or a part of this representation). A unary rule picks up a (sub)tree on the basis of its own properties, thus implicitly establishing a preference relation between this (sub)tree and all its competitors. Each preference rule - be it binary or unary - is associated with a score, which is assigned to the preferred (sub)tree as a result of the application of the rule.

Correspondences: The notion of correspondence between (sub)trees is central to preference rules of the binary type. A number of definitions of this concept can be envisaged:

i. The correspondence between two (sub)trees is established by the user, who states that some specified contraints hold between parts of them.
ii. A correspondence is only assumed to exist between full parse trees, and the correspondence between two subtrees is defined by specifying their derivation paths from the top node.
iii. The system produces a parse graph which will be a synthesis of the various parse trees, where parts common to several trees are shared; two subtrees correspond if they share a given part.

The most challenging solution is (iii): we have not adopted it because of computational problems connected with the introduction of structure-sharing into the E-framework. The easiest solution to implement is (ii): this is the approach chosen in the earlier Eurotra preference tool. The solution we have adopted is (i), which unlike (ii) allows the user to state constraints on subtrees, regardless of their position in the complete parse tree. In other words, our system allows for very local and modular statements.

Preference Rules: The user expresses preference statements through a set of binary or unary preference rules (p-rules).

The syntax for a binary rule is

RuleName (*Score*) =
LHS >= *RHS*
where
Annotations .

where:
- *RuleName* is a unique identifier used for trace purposes;
- *Score* is a positive integer which indicates how strong the relation of preference is;
- *LHS* and *RHS* (the left-hand side and the right-hand side of the rule) are the descriptions of the two (sub)trees to be compared;
- >= is a preference sign that indicates which of the two (sub)trees is to be preferred;
- *Annotations* is a (possibly empty) set of constraints which must hold between the constituents of the two (sub)trees to be compared.

The syntax for a unary rule is

RuleName (*Score*) =
LHS
where
Annotations .

where:
- *LHS* is the description of the (sub)tree to be singled out (which we call the left-hand side to stress the parallelism with binary rules);
- the other parts are as defined for binary rules.

LHS and *RHS* are (sub)tree descriptions of any depth and relevant parts of them may be labelled with Prolog variables, called indexes. These labels are used to express simple or complex correspondence constraints in the annotation part of the rule. A simple constraint states for instance that two indexed subtrees must or must not have the same structure. Simple constraints may be combined with the operators 'and' and 'or' to form complex constraints. Scores, which have the function of driving p-rule interaction, are positive integers. They may be either assigned by the user or generated automatically on statistical grounds, as explained below. Examples of both rule types are given in the appendix.

General Algorithm: All the parse trees have an initial null score before preference rules are applied. For each pair of trees, if they contain two subtrees respectively matching the *LHS* and the *RHS* of a binary p-rule, while the constraints in the annotation part of the rule hold, the rule applies. Similarly, for each parse tree, if a subtree matching the *LHS* of a unary b-rule can be extracted, and all the constraints expressed in the rule are satisfied, the rule applies. In both cases, as a result of p-rule application, the score of the object that contains the preferred subtree is incremented by the score of the rule.

When all binary rules have been tried out on all the possible pairs of trees in all the possible ways, and all unary rules have been fired on all the single trees, results are collected. All parse trees are partitioned into equivalence classes according to their score. Note that trees to which no preference rule has applied will belong to the lowest-ranking class: this is motivated by the assumption that unary rules prefer single trees over all the other members of the set of competing trees.

After this partial order has been established, all the trees but those belonging to the highest-ranking class are discarded.

A possible enhancement to the expressive power of p-rules would be the introduction of negative scores, for cases where a p-rule describes an acceptable but not totally correct subtree.

AN EXAMPLE

The following set of p-rules are based on some of the criteria for the treatment of PP attachment described in [Hirst 1987]. Note that p-rule scores have been assigned manually, due to the small number of rules.

pmod (8) = {cat=pp, sf~=mod} [P1:{cat=p}, NP1:{cat=np}]

>= {cat=pp, sf=mod} [P2:{cat=p}, NP2:{cat=np}]

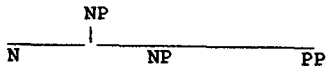
where P1=P2, NP1=NP2.

In the rule above, {} delimit a node in the tree, which in the E-framework is a set of attribute value pairs, [] following a given node enclose its daughters, = means equal to and ~= means different from. The rule prefers a valency-bound PP to a PP modifier. This is a very strong criterion, which can only be overridden by semantic principles: therefore, the rule has a high score.

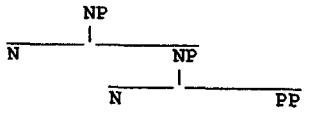
```
plow (2) = {cat=np} [ {cat=n},
                      *{},
                      ## {cat=pp},
                      *{} ].
```

The rule gives 2 points to an attachment where a PP is placed under an NP node. Note that *{} means any number of (sub)trees, without any restriction, and ## in front of a subtree means that this subtree is weakly dominated by the top node. Assuming the following two structures

a)



b)



'plow' will only apply once to (a), but it would fire twice on (b), which will in the end collect the highest score. The rule implements in fact the principle of right-low attachment.

```
pcoord (5) = {cat=?}
    [C1: {sf=conjunct},
     C2: {sf=conjunct}]

where width(C1) = width(C2).
```

The rule above assigns 5 points to a coordinated structure where the two conjuncts have the same number of terminals. Note that constraints are stated between nodes of two competing (sub)trees and not, as it was the case in 'pmod', between nodes belonging to the same (sub)tree.

To see how these p-rules work, we can apply them to the set of objects resulting from the analysis of the following three Danish sentences:

(1) "Kommissionen diskuterede et forslag fra virksomhederne om effektiv løsning af problemerne".

(EN: The commission discussed a proposal by the companies for the effective solution of the problems).

(2) "Virksomhederne deltager i programmet for denne periode".

(EN: The companies take part in the programme for this period).

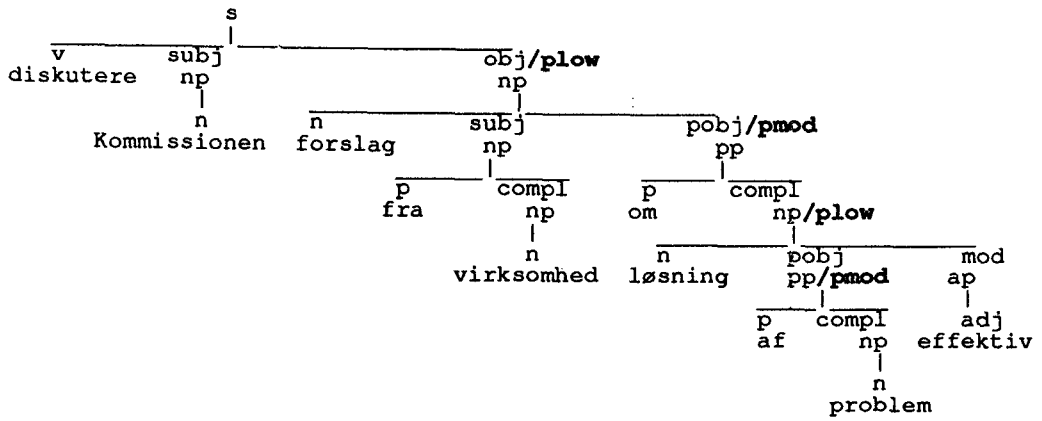
(3) "Kommissionen kontrollerer finansieringen af virksomhederne og samarbeidet med industrien".

(EN: The commission controls the financing of the firms and the cooperation with industry).

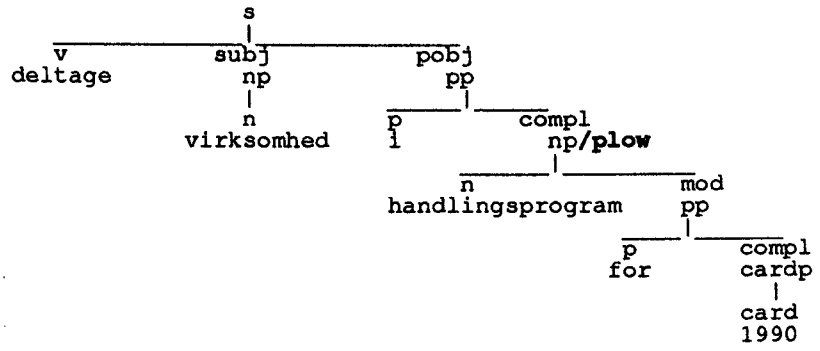
In all three cases the preference tool yields the correct result. The three preferred objects are shown below: p-rules that have applied are indicated on the top nodes of the relevant subtrees.

In accordance with the Eurotra linguistic model, object 1 and 2 below are dependency structures with a lowered governor, where the complements have been ordered in a canonical way and a series of phenomena (determinateness, verbal inflection, prepositions) have been featurised. What interests us here, however, is the way PPs have been analysed. Thus, note that for all the PPs in sentence (1), the system has been able to find valency-bound syntactic functions (either subject or prepositional object).

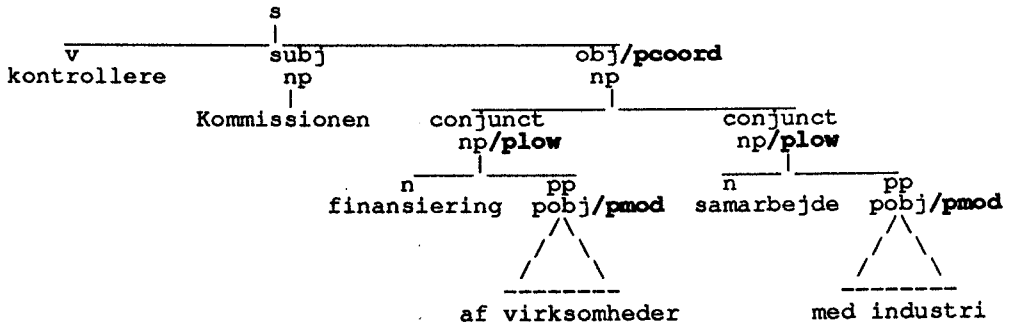
Obj1:



Obj2:



Obj3:



Consequently, modifier interpretations have been dispreferred. In the case of sentence (2) instead, the final PP has been analysed as a modifier, and the correct attachment has been found by the principle of right-low attachment. Note also that, still in (2), the verb "deltage" requires an obligatory prepositional object, and therefore this syntactic function has not been established by preference. Finally, in (3) the correct attachment of the two PPs has been found due to the combined effect of all three rules.

SCORING

Scoring is an important novelty proposed in our system to replace the rule ordering strategy in use in the previous Eurotra preference tool. Whereas arbitrary decisions were made in the earlier tool in cases of contradictory preference criteria and multiple matches between a rule and two (sub)trees, scoring permits us to control the interaction of preference rules in a declarative way. However, there is a tradeoff between the declarativeness permitted by a scoring system and the difficulty of finding the right scores for a p-rule set of nontrivial coverage.

In this section we show how optimum p-rule scores can be derived automatically. Starting from a set of p-rules and an initial set of objects ordered by the user, the system tries to compute optimum values for the p-rules in the set, on the assumption that they will hold for different sets of objects.

If p_i (i=1,...n) stands for the score of the i-th p-rule, then the j-th object is assigned a score S_j given by the following expression:

(1) $S_j = p_1a_{1j} + p_2a_{2j} + ... + p_na_{nj}$ (j=1,...N)

where n is the number of existing p-rules, a_{ij} is a constant equal to the number of times p-rule i has applied to object j and N is the number of existing objects. In other words, S_j stands for the final score totalled by a given object after all possible p-rules have applied to it as many times as possible.

To compute optimum scores, an arbitrary high score is assigned to the best object(s) in the initial training corpus and a much lower one to the rest. The set of equations (1) is transformed then into an overdetermined system of N equations with n unknowns - the p-rule scores - where N can be greater than n. The set of equations (1) can be further decomposed and reformulated as follows:

Find x_i (i=1,...n+1) such that,

(2) $x_1a_{1j} + x_2a_{2j} + ... + x_na_{nj} - x_{n+1}S_j = 0$
(j=1,...N)

By comparing the set of equations (2) against the set (1), the following relation between the values of x_i and p-rule scores is deduced:

(3) $p_i = x_i/x_{(n+1)}$

Therefore, we claim that problems (1) and (2) are equivalent. Now, problem (2) has no exact solution whenever N is greater than n. However, it can be solved by converting it into a constraint optimization problem whereby optimum scores for p-rules will emerge. Thus the set of equations (2) is rearranged by introducing the errors e_j (j=1,...N) and by imposing that the sum of all these errors is minimum. More precisely problem (2) takes now the following form:

Find x_i (i=1,...n+1) such that

$e_{12} + e_{22} + ... + e_{N2}$ ---> minimum

subject to the constraints

(4) $e_j = x_1a_{1j} + x_2a_{2j} + ... + x_na_{nj} - x_{n+1}S_j$ (j=1,...N)

$x_{12} + x_{22} +... x_{(n+1)2} = 1$

In the literature (cf. [Key & Marple 1981] and [Kumaresan & Tufts 1982]), one of the most efficient techniques offered to the solution of the constraint optimization problem (4) is called Singular Value Decomposition (SVD). SVD provides an optimum set of x_i (i=1,...n+1) which guarantees minimum accumulated squared error. Thus the values of the scores p_i (i=1,...n) are computed in a straightforward way from the x_i (i=1,...n+1) using equation (3).

Note that SVD is a non-linear optimization technique which provides the best set of parameters for a given training corpus. Therefore, it is important to apply it to a linguistically balanced corpus. Moreover, for the produced result to be reliable, the existing number of equations N should be at least five to ten times bigger than the existing number of p-rules n.

Although SVD provides an optimum set of p-rule scores, there is no guarantee that these scores are all positive. However, since p-rules express positive selection criteria, p-rule scores must always be positive: the following paragraph proposes an iterative algorithm which computes p-rule scores guaranteeing their positiveness at the same time.

The idea is that the set of SVD parameters x_i (i=1,...n+1) and the N sets of parameters in the training corpus are uncorrelated sets, i.e. they do not belong to the same space section. If the SVD solution set x_i (i=1,...n+1) is also included in the training set, the new SVD solution y_i (i=1,...n+1) of the augmented training corpus will be uncorrelated to all the sets in the corpus. Consequently, y_i (i=1,...n+1) will also be uncorrelated to x_i (i=1,...n+1). This means that not all the signs of y_i (i=1,...n+1) will be identical to the signs of x_i (i=1,...n+1). If the y components are all positive or all negative, the algorithm ends successfully and positive p-rule scores are computed via equation (3). In all other cases, the set of y_i (i=1,...n+1) is also incorporated in the training corpus and a new SVD solution z_i (i=1,...n+1) is computed which is uncorrelated to both x_i and y_i (i=1,...n+1). The algorithm continues in a similar way by checking whether the signs of z_i (i=1,...n+1) are all the same or not: in the first case the algorithm ends successfully; in the second case the set of z_i (i=1,...n+1) is included in the corpus and a new SVD solution is computed.

The algorithm will eventually come up with the desirable set of parameters when all alternatives have been exhausted throughout the preceding iterations. The time of convergence varies relative to the number of parameters or, equivalently, to the number of p-rules, as well as the size of the training corpus. More precisely, the larger the number of p-rules, the longer it takes for the algorithm to converge. On the other hand, the larger the training corpus, the faster the time of convergence. The obtained solution is optimum given the imposed constraint that all p-rule scores are positive.

CONCLUSION

It seems to us that two basic tendencies can be observed in the literature with respect to the treatment of preference. On the one hand, preference is conceived of as an essentially linguistic or psycholinguistic principle or sum of principles (cf. the LFG approach in [Ford et al. 1982]); although it has important consequences for the parser, preference is not directly connected to a specific parsing method. On the other hand, preference has been studied in the context of parsing: in such treatments (cf. [Pereira 1985]), preference amounts to a deterministic procedure, which is not necessarily motivated by linguistic evidence. In our approach preference is established on the basis of rules defined by the user and applied by a post-processor. We have in fact focussed on a method to express linguistically meaningful preference statements rather than on a particular parsing strategy. We are aware of the fact that, in a system where parsing is seen as a constraint satisfaction problem, preference criteria of the type we are interested in can be treated on the same level as other linguistic constraints and used to resolve ambiguity at parse time (cf. [Van Henteryck 1989]). However, such an approach would have meant too radical a change to the underlying Eurotra formalism.

In accordance with the general practice in Eurotra, our preference mechanism does not plead allegiance to any specific linguistic theory. We have, however, been influenced by a theoretical framework, namely the theory of preference rule systems described in [Jackendoff 1985]. According to this framework, preference can only been decided on the basis of a number of criteria, and a preference mechanism is not based on a dichotomy between correct and wrong results, but on a scale of degrees of acceptability. One of our main concerns in designing the system, in fact, has been allowing various and even contradictory criteria to be combined in a declarative fashion. The use of scoring is in this sense crucial.

The system has been implemented and successfully tested on real input which showed overgeneration due to PP and adverbial attachment, coordination, pronominal resolution and lexical ambiguity. Some testing results are given in the appendix.

ACKNOWLEDGEMENTS

This work has been carried out by a group consisting of Paul Bennett (Eurotra-GB, Umist), Dieter Maas (EurotraDE, Saarbruecken), Juan Carlos Ruiz (Eurotra-ES, Barcellona) and the authors of the present paper.

We thank Bolette Pedersen (Eurotra-DK) who contributed to the formulation of the p-grammar for Danish.

APPENDIX

TEST NUMBER				2
NO. OF SENTENCES				30
AVERAGE SENTENCE LENGTH IN WORDS				9.74
grammar type	no. of p-rules	average analyses no. per sentence	average cpu per sentence	correct results
with p-rules	21	1.7	436s	93.3 %
without p-rules	0	16	333s	irrelevant

Fig. 1.

Figure 1 shows the results obtained in a test carried out by Eurotra-IT (Dima group). The linguistic phenomena handled by p-rules included syntactic completeness check, ambiguity of semantic role assignment for arguments, ambiguity of semantic labelling for modifiers. The experiment was performed on a Sparkstation I (16 MB core memory)

REFERENCES

A.Bech, B.Maegaard & A.Nygaard (1990), "The EUROTRA MT Formalism", forthcoming in Machine Translation, ed. Sergei Nirenburg.

D.Fass & Y.Wilks (1983), "Preference Semantics, Ill-Formedness, and Metaphor", in Americal Journal of Computational Linguistics, vol. 9, no.3-4, July--Dec.

M.Ford et al. (1982), "A Competence-Based Theory of Syntactic Closure", in J.Bresnan ed., The Mental Representation of Grammatical Relations, The MIT Press: Cambridge Mass.

L.Frazier & J.D.Fodor (1978), "The Sausage Machine: A New Two-Stage Parsing Model", in Cognition, vol.6.

Jackendoff (1985), Semantics and Cognition, Mit Press: Cambridge, Mass.

G.Hirst (1987), Semantic Interpretation and the Resolution of Ambiguity, Cambridge University Press: Cambridge.

X.Huang (1988), "Semantic Analysis in XTRA, An English-Chinese Machine Translation System", in Computers and Translation, vol.3, no.2.

S.M.Key & S.L.Marple (1981), "Spectrum Analysis - A Modern Perspective", in Proceedings of the IEEE, Vol. 69.

R.Kumaresan & D.Tufts (1982), "Singular Value Decomposition and improved Frequency Estimation Using Linear Prediction", IEEE ASSP, vol.30, no. 4.

G.Malnati & P.Paggio (1990), "The Eurotra User Language", forthcoming in Machine Translation and Natural Language Processing, vol.2, CEC, Luxembourg.

P.Paggio (1988), "The Concept of Preference Applied to the Automatic Analysis of PPs and ADVPs", in SAML, Copenhagen.

F.C.N.Pereira (1985), "A New Characterization of Attachment Preferences", in Dowty et al., Natural Language Parsing, Cambridge University Press: Cambridge.

D.Petitpierre et al. (1987), "A Model for Preference", in Proceedings of the Third ACL Conference, Copenhagen.

A.Raw et al. (1989), An Introduction to the Eurotra Machine Translation System, in Working Papers in Natural Language Processing, no.1, Leuven.

P.Van Henteryck (1989), Constraint Satisfaction in Logic Programming, The MIT Press: Cambridge Mass.

Y.Wilks & A.Herscovits (1977), "An Intelligent Analyser and Generator for Natural Language", in Computational and Mathematical Linguistics, Leo S.Olschki Ed: Firenze.