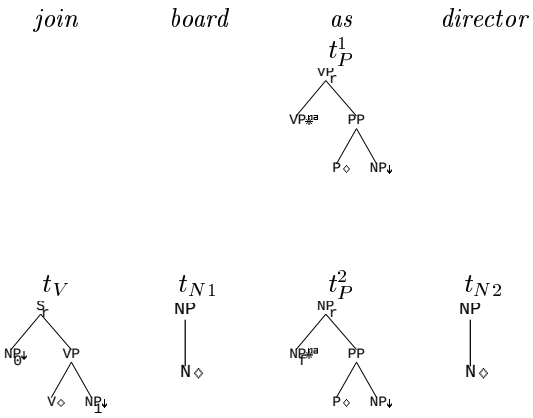
1 Corpus-based Parsing of TAGs

A promising approach in making the parsing of TAGs more efficient is to exploit distributional evidence from corpora in order to select the most plausible path through the search space of possible derivations for a sentence. We describe here a technique for parsing that exploits structural relations assigned by a lexicalized TAG grammar and a probability model of lexical preference trained from unannotated text (cite [HindleandRooth]).

Structural relations are assigned to words as trees from the XTAG wide-coverage lexicalized grammar. For each word in a sentence these trees encode argument and adjunct relations in various syntactic environments. Parsing can now be viewed as computing attachments for these relations.

To illustrate the technique we conducted an experiment considering the case of computing PP-attachments while parsing. In the following example, we have to disambiguate which of the two attachment possibilities presented by the two trees lexicalized by the preposition *as* is the more plausible one.



We can make this decision by considering the following model of attachments, where the event of an adjunction of a tree t at a node labelled X is denoted by $X \mapsto t$. Also the event that no adjunction occurs at a node labelled X is denoted by $X \mapsto nil$. A node labelled X in a tree lexicalized by a word w is written as X_w.

$$\Pr(\text{VP attach P} \mid V, N1, P, N2) = \Pr(VP_V \mapsto t_P^1 \mid P, V, N2) \quad (1)$$

$$\Pr(\text{VP attach } nil \mid V, N1, P, N2) = \Pr(VP_V \mapsto nil) \quad (2)$$

$$\Pr(\text{NP attach P} \mid V, N1, P, N2) = \Pr(NP_{N1} \mapsto t_P^2 \mid P, N1, N2) \quad (3)$$

$$\Pr(\text{NP attach } nil \mid V, N1, P, N2) = \Pr(NP_{N1} \mapsto nil) \quad (4)$$

We can now choose between *VP attach P* and *NP attach P* by taking the log odds. Since in this experiment we assume we are given exactly quadruples

of V, N1, P, N2 we can ignore the terms Pr(VP attach nil | $V, N1, P, N2$) and Pr(NP attach nil | $V, N1, P, N2$).

$$LA = log_2 \frac{\Pr(\text{VP attach P} \mid V, N1, P, N2)}{\Pr(\text{NP attach P} \mid V, N1, P, N2)} \quad (5)$$

The value for LA gives us a confidence measure for how well we can decide the attachment. A *threshold* value tells us when to ignore the value for LA and attach using the default strategy (usually attach low). The *threshold* value is set by experimenting on the development test set. We make the following approximation to allow us to estimate from unannotated data:

$$\Pr(VP_V \mapsto t^1_P \mid P, V, N2) \approx \hat{p}(P \mid V, N2) \quad (6)$$
$$\Pr(NP_{N1} \mapsto t^2_P \mid P, N1, N2) \approx \hat{p}(P \mid N1, N2) \quad (7)$$

These are estimated by taking the maximum likelihood estimate and backing off when there are low counts. The backing off strategy is adapted from [MikeCollinsPPpaper]. We use a variable cutoff level c for the first level of backoff.

In order to evaluate this approach we took a set of ambiguous prepositional attachment cases from the Penn Treebank [citeTreeBank.paper] (this data was extracted by [cite IBM.paper]). We stripped the attachment decision from the training data and after training our model, tested on the development test set. The number of training examples was 20801 and the number of test examples was 4039.

We assume that *of* lexicalizes only the tree corresponding to NP attachment (i.e. *of* only takes $\beta nxPnx1$). The baseline accuracy for this task is 53% if you always attach low (to the NP).

threshold	percent correct
0	0.747
0.2	0.759
0.4	0.762
0.6	0.766
0.8	0.769
0.9	0.769
1	0.766
1.5	0.756
2	0.750

Table 1: PP attachment accuracy

The results obtained are comparable to results obtained with the earliest supervised algorithms [cite IBM.paper]. They used the same training and test

data so it is a fair comparision. Not surprisingly, the performance is not as good as current supervised techniques which achieve around 88% accuracy on this data [Stetina.Nagao]. When compared to other unsupervised techniques for the same task (on the same test data) this algorithm compares favorably. [Adwait.ACLtoappear] performs at around 78% for the same task using an unsupervised maximum entropy model which is very close to the 77% we achieve here. There are some differences in training data which make the results not directly comparable. [Adwait.ACLtoappear] creates quadruples of PP attachment cases by heuristically searching through a part of speech tagged corpus, hence that result is based on a noisy but very large (24M words) corpus whereas our algorithm is trained on a much smaller training corpus (20K words) but where quadruples are never spurious. We plan to extend this basic approach by incorporating better smoothing techniques including backing off to the part of speech labels of the words or backing off to automatically induced classes of words [Brown.etal].

As we pointed out before the objective of this experiment was to test a model of attachments for TAGs which in turn disambiguates the structure assigned to a sentence. Since we couched the attachments decisions as adjunctions and substitutions, we can now extend this model to the disambiguation of all substitutions and adjunctions in a TAG parser. The backoff strategy used here also extends in a straightforward way. While here we always backed off relative to the preposition, in a full TAG parser we always back off relative to the anchor of the tree being attached. We are currently working on incorporating our technique of PP attachment to general parsing using a TAG grammar.