New Figures of Merit for Best-First Probabilistic Chart Parsing *

Sharon A. Caraballo and Eugene Charniak
Brown University
{sc,ec}@cs.brown.edu

October 24, 1997

Abstract

Best-first parsing methods for natural language try to parse efficiently by considering the most likely constituents first. Some figure of merit is needed by which to compare the likelihood of constituents, and the choice of this figure has a substantial impact on the efficiency of the parser. While several parsers described in the literature have used such techniques, there is little published data on their efficacy, much less attempts to judge their relative merits. We propose and evaluate several figures of merit for best-first parsing, and we identify an easily-computable figure of merit which provides excellent performance on various measures and two different grammars.

1 Introduction

Chart parsing is a commonly-used algorithm for parsing natural language texts. The chart is a data structure which contains all of the constituents for which subtrees have been found, that is, constituents for which a derivation has been found and which may therefore appear in some complete parse of the sentence. The agenda is a structure that stores a list of constituents for which a derivation has been found but which have not yet been combined with other constituents. Initially, the agenda contains the terminal symbols from the sentence to be parsed. A constituent is removed from the agenda and added to the chart, and the system considers how this constituent can be used to extend its current structural hypothesis by combining with other constituents in the chart according to the grammar rules. (We will often refer to these expansions of rules as

*This research was supported in part by NSF grant IRI-9319516 and by ONR grant N0014-96-1-0549.

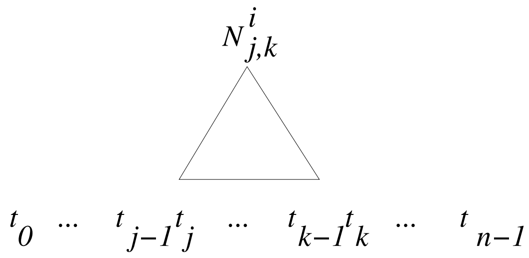

Figure 1: Constituent $N^i_{j,k}$ in a sentence $t_{0,n}$

"edges".) In general this can lead to the creation of new, more encompassing constituents which themselves are then added to the agenda. When one constituent has been processed, a new one is chosen to be removed from the agenda, and so on. Traditionally, the agenda is represented as a stack, so that the last item added to the agenda is the next one removed. Chart parsing is described extensively in the literature; for one such discussion see Section 1.4 of [5].

Best-first probabilistic chart parsing is a variation of chart parsing which attempts to find the most likely parses first, by adding constituents to the chart in order of the likelihood that they will appear in a correct parse, rather than simply popping constituents off of a stack. Some probabilistic figure of merit is assigned to the constituents on the agenda, and the constituent maximizing this value is the next to be added to the chart.

In this paper we consider probabilities primarily based on probabilistic context-free grammars, though in principle other, more complicated schemes could be used.

The purpose of this work is to evaluate how well several figures of merit select constituents to be moved from the agenda to the chart. Ideally, we would like to use as our figure of merit the conditional probability of that constituent, given the entire sentence, in order to choose a constituent that not only appears likely in isolation, but is most likely given the sentence as a whole; that is, we would like to pick the constituent that maximizes the following quantity:

$$p(N^i_{j,k}|t_{0,n})$$

where $t_{0,n}$ is the sequence of the n tags, or parts of speech, in the sentence (numbered $t_0, \ldots, t_{n-1}$), and $N^i_{j,k}$ is a nonterminal of type i covering terms $t_j \ldots t_{k-1}$. (See Figure 1.)

In our experiments, we use only tag sequences (as given in the test data) for parsing. More accurate probability estimates should be attainable using lexical information in future experiments, as more detail usually leads to better statistics, but lexicalized figures of merit are beyond the scope of the research described here.

Note that our "ideal" figure is simply a heuristic, since there is not guarantee that a constituent which scores well on this measure will appear in the correct parse of a sentence. For example, there may be a very large number of low-probability derivations of $N^i_{j,k}$, which are combined here to give a high value, but a parse of the sentence can only include one of these derivations, making it unlikely that $N^i_{j,k}$ appears in the most probable parse of the sentence. On the other hand, there is no reason to believe that such cases are common in practice.

However, we cannot calculate $p(N^i_{j,k}|t_{0,n})$, since in order to do so, we would need to completely parse the sentence. In this paper, we examine the performance of several proposed figures of merit that approximate it in one way or another, using two different grammars. We identify a figure of merit that gives superior results on all of our performance measures and on both grammars.

Section 2 of this paper describes the method we used to determine the effectiveness of figures of merit, that is, to compare how well they choose constituents to be moved from the agenda to the chart. Section 2.1 explains the experiment, Section 2.2 describes the measures we used to compare the performance of the figures of merit, and Section 2.3 describes a model we used to represent the performance of a traditional parser using a simple stack as an agenda.

In Section 3, we describe and compare three simple and easily-computable figures of merit based on inside probability. Sections 3.1 through 3.3 describe each figure in detail, and Section 3.4 presents the results of an experiment comparing these three figures. Sections 4 and 5 have a similar structure to Section 3, with Section 4 evaluating two figures of merit using statistics on the left-side context of the constituent, and Section 5 evaluating three additional figures of merit using statistics on the context on both sides of the constituent. Section 6 contains a table summarizing the results from Sections 3, 4, and 5.

In Section 7, we use another grammar in the experiment, to verify that our results are not an artifact of the grammar used for parsing.

Section 8 describes previous work in this area, and Section 9 presents our conclusions and recommendations.

There are also three appendices to this paper. Appendix A gives our method for computing inside probability estimates while maintaining parser speed. Appendix B explains how we obtained our boundary statistics used in Section 5. Appendix C presents data comparing the parsing accuracy obtained by each of our parsers as the number of edges they create increases.

2 Comparing figures of merit

2.1 The Experiment

We used as our first grammar a probabilistic context-free grammar learned from the Brown corpus (see [9] for a description of the Brown Corpus, and [3], [4], and [7] for grammar and training details). This grammar contains about 5,000 rules using 32 terminal and nonterminal symbols. We parsed 500 sentences of length 3 to 30 (including punctuation) from the Penn Treebank Wall Street Journal corpus [14] using a best-first parsing method and various estimates for $p(N^i_{j,k}|t_{0,n})$ as the figure of merit.

For each figure of merit, we compared the performance of best-first parsing using that figure of merit to exhaustive parsing. By exhaustive parsing, we mean continuing to parse until there are no more constituents available to be added to the chart. We parse exhaustively to determine the total probability of a sentence, that is, the sum of the probabilities of all parses found for that sentence.

We then computed several quantities for best-first parsing with each figure of merit at the point where the best-first parsing method has found parses contributing at least 95% of the probability mass of the sentence. The 95% figure is simply a convenience; see Appendix C for a discussion of speed versus accuracy.

2.2 Measures used

We compared the figures of merit using the following measures:

1. %E: The percentage of edges, or rule expansions, in the exhaustive parse that have been used by the best-first parse to get 95% of the probability mass. Edge creation is a good measure of CFG parser effort since it is independent of platform and implementation.

2. %non-0 E: The percentage of nonzero-length edges used by the best-first parse to get 95%. Zero-length edges are required by our parser as a book-keeping measure, and as such virtually cannot be eliminated. We anticipated that removing them from consideration would highlight the "true" differences in the figures of merit.

3. %popped: The percentage of constituents in the exhaustive parse that were used by the best-first parse to get 95% of the probability mass. This measure was included to confirm that a figure of merit which is efficient in terms of edge creation is also efficient in terms of constituent creation.

4. CPU time: The total CPU time (in seconds) needed to get 95% of the probability mass for all of the 500 sentences.

The statistics converged to their final values quickly. The edge-count percentages were generally within .01 of their final values after processing only 200 sentences, so the results were quite stable by the end of our 500-sentence test corpus.

We gathered statistics for each sentence length from 3 to 30. Sentence length was limited to a maximum of 30 because of the huge number of edges that are

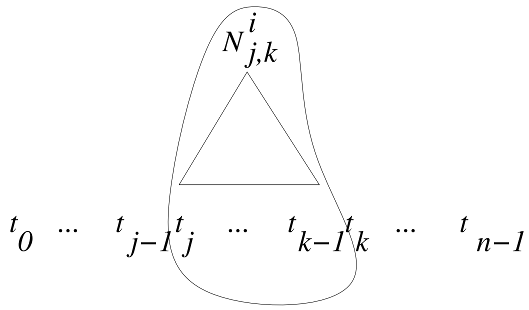

Figure 2: β includes only words within the constituent

generated in doing a full parse of long sentences; using this grammar, sentences in this length range have produced up to 130,000 edges.

2.3 The "stack" model

As a basis for comparison, we measured the CPU time for a non-best-first version of the parser to completely parse all 500 sentences. The CPU time needed by this version of the parser was 4882 seconds. For a best-first version of the parser to be useful, it must be able to find the most probable parse (or a reasonably good parse, depending on the application) in less than this amount of time. Here, for the best-first parsers, we will use for convenience the time needed to get 95% of the sentence's total probability mass.

3 Simple figures of merit

3.1 Straight β

It seems reasonable to base a figure of merit on the inside probability β of the constituent. Inside probability is defined as the probability of the words or tags in the constituent given that the constituent is dominated by a particular nonterminal symbol. (See Figure 2.) This seems to be a reasonable basis for comparing constituent probabilities, and has the additional advantage that it is easy to compute during chart parsing. Appendix A gives details of our online computation of β.

The inside probability of the constituent $N^i_{j,k}$ is defined as

$$\beta(N^i_{j,k}) \equiv p(t_{j,k}|N^i)$$

where N^i represents the ith nonterminal symbol.

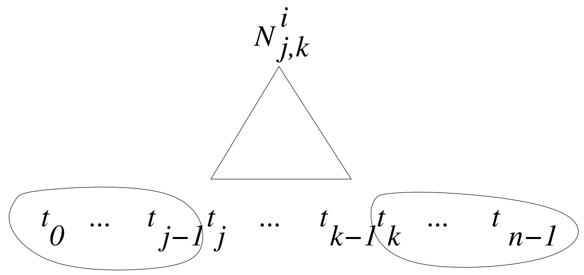

Figure 3: α includes the entire context of the constituent

In terms of our earlier discussion, our "ideal" figure of merit can be rewritten as:

$$\begin{aligned} p(N^i_{j,k}|t_{0,n}) &= \frac{p(N^i_{j,k}, t_{0,n})}{p(t_{0,n})} \\ &= \frac{p(N^i_{j,k}, t_{0,j}, t_{j,k}, t_{k,n})}{p(t_{0,n})} \\ &= \frac{p(t_{0,j}, N^i_{j,k}, t_{k,n})p(t_{j,k}|t_{0,j}, N^i_{j,k}, t_{k,n})}{p(t_{0,n})}. \end{aligned}$$

We apply the usual independence assumption that given a nonterminal, the tag sequence it generates depends only on that nonterminal, giving

$$\begin{aligned} p(N^i_{j,k}|t_{0,n}) &\approx \frac{p(t_{0,j}, N^i_{j,k}, t_{k,n})p(t_{j,k}|N^i_{j,k})}{p(t_{0,n})} \\ &= \frac{p(t_{0,j}, N^i_{j,k}, t_{k,n})\beta(N^i_{j,k})}{p(t_{0,n})}. \end{aligned}$$

The first term in the numerator is just the definition of the outside probability α of the constituent. Outside probability α of a constituent $N^i_{j,k}$ is defined as the probability of that constituent and the rest of the words in the sentence (or rest of the tags in the tag sequence, in our case). (See Figure 3.)

$$\alpha(N^i_{j,k}) \equiv p(t_{0,j}, N^i_{j,k}, t_{k,n}).$$

We can therefore rewrite our ideal figure of merit as

$$p(N^i_{j,k}|t_{0,n}) \approx \frac{\alpha(N^i_{j,k})\beta(N^i_{j,k})}{p(t_{0,n})}.$$

In this equation, we can see that $\alpha(N^i_{j,k})$ and $p(t_{0,n})$ represent the influence of the surrounding words. Thus using β alone assumes that α and $p(t_{0,n})$ can be ignored.

We will refer to this figure of merit as **straight** β.

3.2 Normalized β

One side effect from omitting the α and $p(t_{0,n})$ terms in the straight β figure above is that inside probability alone tends to prefer shorter constituents to longer ones, as the inside probability of a longer constituent involves the product of more probabilities. This can result in a "thrashing" effect as noted in [8], where the system parses short constituents, even very low probability ones, while avoiding combining them into longer constituents. To avoid thrashing, some technique is used to normalize the inside probability for use as a figure of merit. One approach is to take the geometric mean of the inside probability, to obtain a "per-word" inside probability. (In the "ideal" model, the $p(t_{0,n})$ term acts as a normalizing factor.)

The per-word inside probability of the constituent $N^i_{j,k}$ is calculated as

$$\sqrt[k-j]{\beta(N^i_{j,k})}.$$

We will refer to this figure as **normalized** β.

3.3 Trigram estimate

An alternative way to rewrite the "ideal" figure of merit is as follows:

$$\begin{aligned} p(N^i_{j,k}|t_{0,n}) &= \frac{p(N^i_{j,k}, t_{0,n})}{p(t_{0,n})} \\ &= \frac{p(t_{0,j}, t_{k,n})p(N^i_{j,k}|t_{0,j}, t_{k,n})p(t_{j,k}|N^i_{j,k}, t_{0,j}, t_{k,n})}{p(t_{0,j}, t_{k,n})p(t_{j,k}|t_{0,j}, t_{k,n})}. \end{aligned}$$

Once again applying the usual independence assumption that given a nonterminal, the tag sequence it generates depends only on that nonterminal, we can rewrite the figure of merit as follows:

$$p(N^i_{j,k}|t_{0,n}) \approx \frac{p(N^i_{j,k}|t_{0,j}, t_{k,n})\beta(N^i_{j,k})}{p(t_{j,k}|t_{0,j}, t_{k,n})}.$$

To derive an estimate of this quantity for practical use as a figure of merit, we make some additional independence assumptions. We assume that $p(N^i_{j,k}|t_{0,j}, t_{k,n}) \approx p(N^i_{j,k})$, that is, that the probability of a nonterminal is independent of the tags before and after it in the sentence. We also use a trigram model for the tags themselves, giving $p(t_{j,k}|t_{0,j}, t_{k,n}) \approx p(t_{j,k}|t_{j-2}, t_{j-1})$. Then we have:

$$p(N^i_{j,k}|t_{0,n}) \approx \frac{p(N^i)\beta(N^i_{j,k})}{p(t_{j,k}|t_{j-2}, t_{j-1})}.$$

We can calculate $\beta(N^i_{j,k})$ as usual.

The $p(N^i)$ term is estimated from our PCFG and the training data from which the grammar was learned. We estimate $p(N^i)$ as the sum of the counts for all rules having N^i as their left-hand side, divided by the sum of the counts for all rules. [1]

The $p(t_{j,k}|t_{j-2}, t_{j-1})$ term is just the probability of the tag sequence $t_j \dots t_{k-1}$ according to a trigram model.(Technically, this is not a trigram model but a tritag model, since we are considering sequences of tags, not words.) Our tritag probabilities $p(t_a|t_{a-2}, t_{a-1})$ were learned from the training data used for the grammar, using the deleted interpolation method for smoothing. Our figure of merit uses

$$p(t_{j,k}|t_{j-2}, t_{j-1}) \approx \prod_{a=j}^{k-1} p(t_a|t_{a-2}, t_{a-1})$$

We refer to this figure of merit as the **trigram estimate**.

3.4 Results

The results for the three figures of merit introduced in the last section according to the measurements given in Section 2.2 are as follows (the time to fully parse using the "stack" model is included for easy reference):

Figure of Merit	%E	%non-0 E	%popped	CPU time
straight β	97.6	97.5	93.8	3966
normalized β	34.7	31.6	61.5	1631
trigram estimate	25.2	21.7	44.3	1547
"stack"	-	-	-	4882

Figure 4 expands the %non-0 E data to show the percent of nonzero-length edges needed to get 95% of the probability mass for each sentence length.

Straight β performs quite poorly on this measure. In order to find 95% of the probability mass for a sentence, a parser using this figure of merit typically needs to do over 90% of the work. On the other hand, normalized β and the trigram estimate both result in substantial savings of work. However, while these two models produce near-equivalent performance for short sentences, for longer sentences, with length greater than about 15 words, the trigram estimate gains a clear advantage. In fact, the performance of normalized β appears to level off in this range, while the amount of work done using the trigram estimate shows a continuing downward trend.

Figure 5 shows the average CPU time to get 95% of the probability mass for each estimate and each sentence length. Each estimate averaged below 1 second

[1] Our results show that the $p(N^i)$ term can be omitted from this figure of merit without much effect.

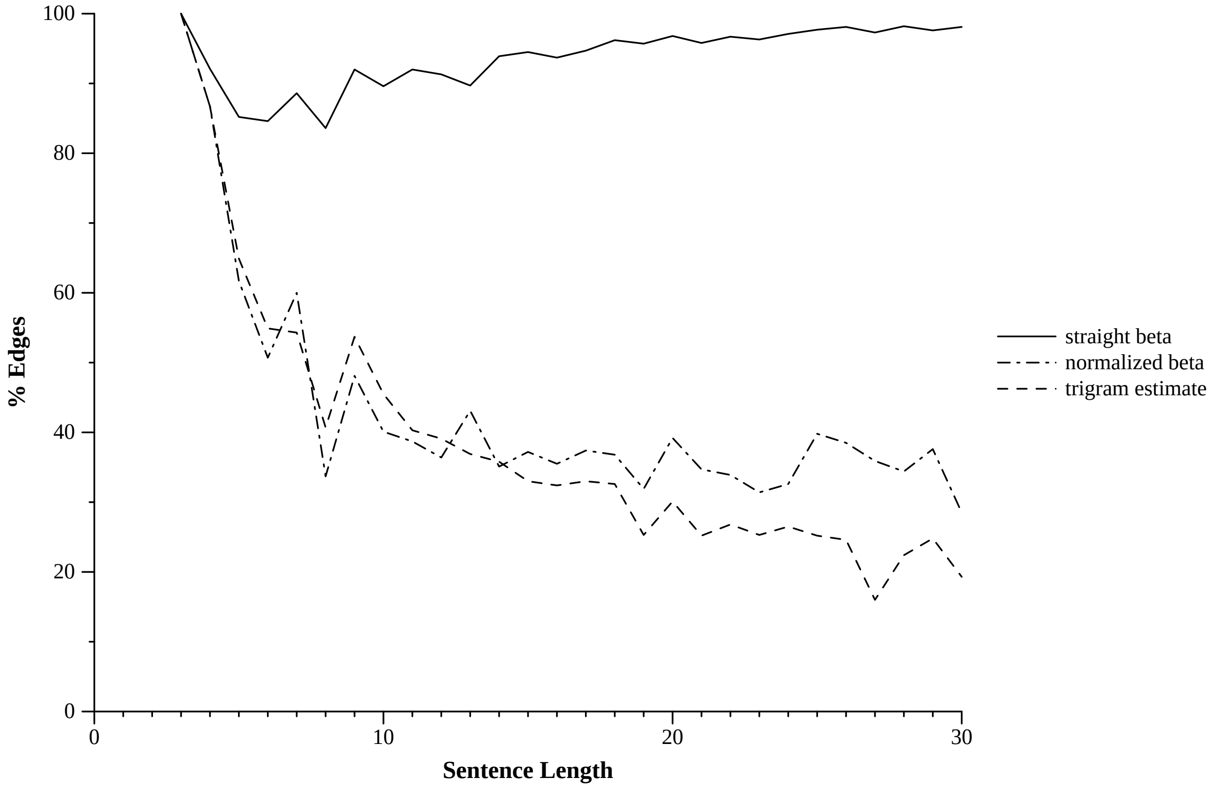

Figure 4: Nonzero-length edges for 95% of Probability Mass for Beta Estimates

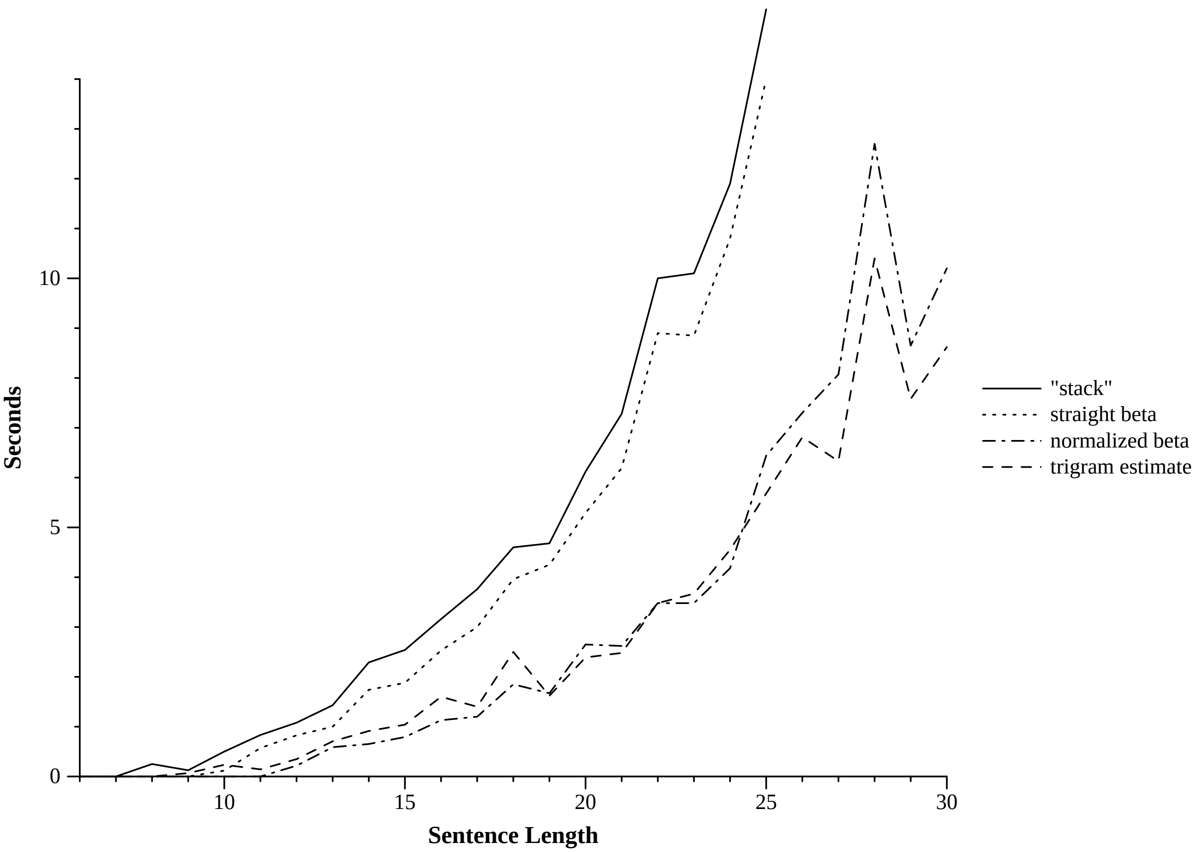

Figure 5: Average CPU Time for 95% of Probability Mass for Beta Estimates

on sentences of fewer than 7 words. (The y-axis has been restricted so that the normalized β and trigram estimates can be better compared.

Note that while straight β does perform better than the "stack" model in CPU time, the two models approach equivalent performance as sentence length increases, which is what would be expected from the edge counts measures. The other two models provide a real time savings over the "stack" model, as can be seen from Figure 5 and from the total CPU times given earlier. Through most of the length range, the CPU time needed by the normalized β and the trigram estimate is quite close, but at the upper end of the range we can see better performance by the trigram estimate. (This improvement comes later than in the edge count statistics because of the small additional amount of overhead work needed to use the trigram estimate.)

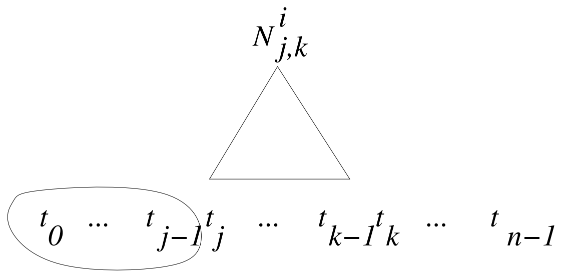

Figure 6: Left outside context

4 Figures involving left outside probability

4.1 Normalized $\alpha_L\beta$

Earlier, we showed that our ideal figure of merit can be written as

$$p(N^i_{j,k}|t_{0,n}) \approx \frac{\alpha(N^i_{j,k})\beta(N^i_{j,k})}{p(t_{0,n})}.$$

However, the α term, representing outside probability, cannot be calculated directly during a parse, since we need the full parse of the sentence to compute it. In some of our figures of merit, we use the quantity $p(N^i_{j,k}, t_{0,j})$, which is closely related to outside probability. We call this quantity the left outside probability, and denote it α_L. (See Figure 6.)

The following recursive formula can be used to compute α_L. Let $\mathcal{E}^i_{j,k}$ be the set of all edges, or rule expansions, in which the nonterminal $N^i_{j,k}$ appears. For each edge e in $\mathcal{E}^i_{j,k}$, we compute the the product of α_L of the nonterminal appearing on the left-hand side (lhs) of the rule, the probability of the rule itself, and β of each nonterminal $N^q_{r,s}$ appearing to the left of $N^i_{j,k}$ in the rule. Then $\alpha_L(N^i_{j,k})$ is the sum of these products:

$$\alpha_L(N^i_{j,k}) \quad = \quad \sum_{e\in\mathcal{E}^i_{j,k}} \alpha_L(N^{\mathrm{lhs}(e)}_{\mathrm{start}(e),\mathrm{end}(e)})p(\mathrm{rule}(e))\prod_{N^q_{r,s}}\beta(N^q_{r,s}).$$

Given a complete parse of the sentence, the formula above gives an exact value for α_L. During parsing, the set $\mathcal{E}^i_{j,k}$ is not complete, and so the formula gives an approximation of α_L.

This formula can be infinitely recursive, depending on the properties of the grammar. A method for calculating α_L more efficiently can be derived from the calculations given in [10].

A simple extension to the normalized β model allows us to estimate the per-word probability of all tags in the sentence through the end of the constituent under consideration. This allows us to take advantage of information already obtained in a left-right parse. We calculate this quantity as follows:

$$\sqrt[k]{\alpha_L(N^i_{j,k})\beta(N^i_{j,k})}.$$

We are again taking the geometric mean to avoid thrashing by compensating for the $\alpha\beta$ quantity's preference for shorter constituents, as explained in the previous section.

We refer to this figure of merit as **normalized** $\alpha_L\beta$.

4.2 Prefix estimate

We also derived an estimate of the ideal figure of merit which takes advantage of statistics on the first $j-1$ tags of the sentence as well as $t_{j,k}$. This estimate represents the probability of the constituent in the context of the preceding tags.

$$\begin{aligned}
p(N^i_{j,k}|t_{0,n}) &= \frac{p(N^i_{j,k}, t_{0,n})}{p(t_{0,n})} \\
&= \frac{p(t_{k,n})p(N^i_{j,k}, t_{0,j}|t_{k,n})p(t_{j,k}|N^i_{j,k}, t_{0,j}, t_{k,n})}{p(t_{k,n})p(t_{0,k}|t_{k,n})} \\
&= \frac{p(N^i_{j,k}, t_{0,j}|t_{k,n})p(t_{j,k}|N^i_{j,k}, t_{0,j}, t_{k,n})}{p(t_{0,k}|t_{k,n})}.
\end{aligned}$$

We again make the independence assumption that $p(t_{j,k}|N^i_{j,k}, t_{0,j}, t_{k,n}) \approx \beta(N^i_{j,k})$. Additionally, we assume that $p(N^i_{j,k}, t_{0,j})$ and $p(t_{0,k})$ are independent of $p(t_{k,n})$, giving

$$p(N^i_{j,k}|t_{0,n}) \approx \frac{p(N^i_{j,k}, t_{0,j})\beta(N^i_{j,k})}{p(t_{0,k})}.$$

The denominator, $p(t_{0,k})$, is once again calculated from a tritag model. The $p(N^i_{j,k}, t_{0,j})$ term is just α_L, defined above in the discussion of the normalized $\alpha_L\beta$ model. Thus this figure of merit can be written as

$$\frac{\alpha_L(N^i_{j,k})\beta(N^i_{j,k})}{p(t_{0,k})}.$$

We will refer to this as the **prefix estimate**.

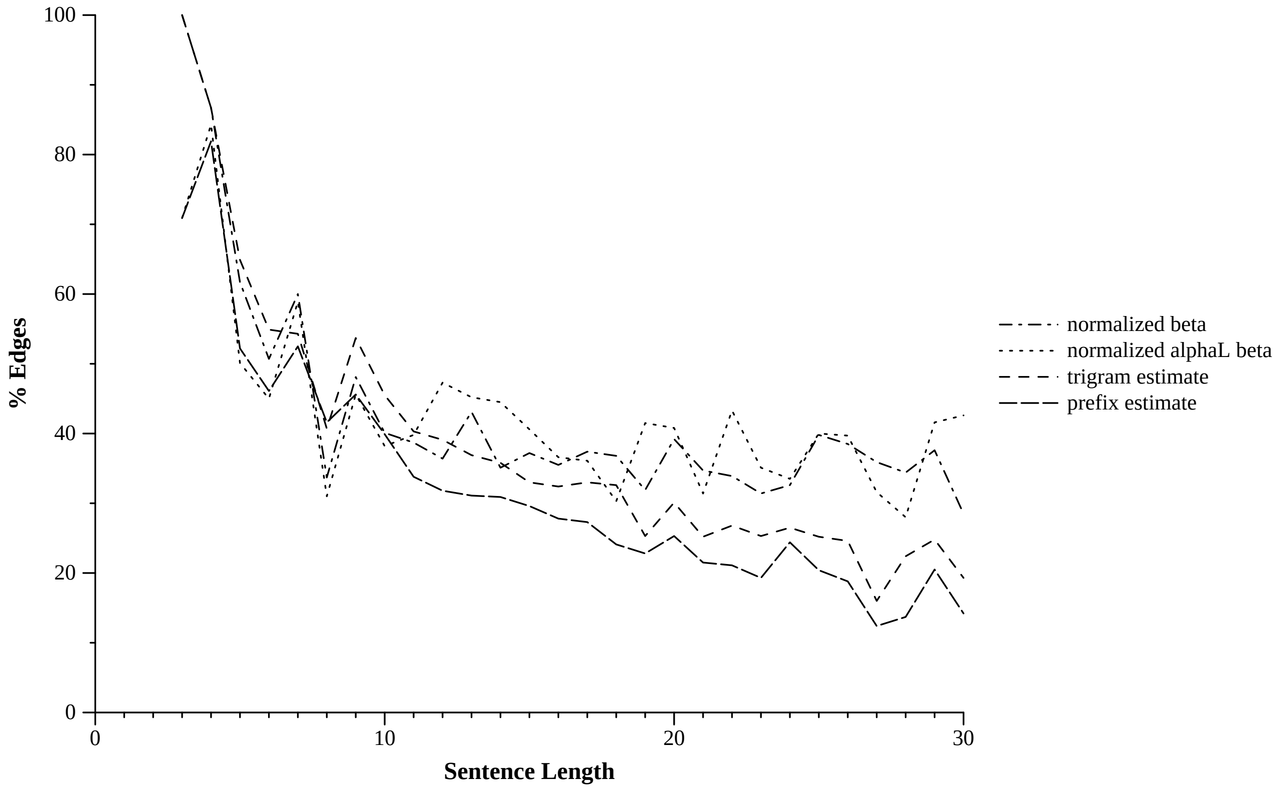

Figure 7: Nonzero-length edges for 95% of Probability Mass for AlphaL Beta Estimates

4.3 Results

The results for the figures of merit introduced in the previous section according to the measurements given in Section 2.2 are as follows:

Figure of Merit	%E	%non-0 E	%popped	CPU time
normalized $\alpha_L\beta$	39.7	36.4	57.3	68660
prefix estimate	21.8	17.4	38.3	26520

Figure 7 shows a graph of %non-0 E for each sentence length for the two α_L models and the related β models.

Figure 7 illustrates two main points. First, the deterioration of the performance of the geometric-mean based models with sentence length can be seen clearly. Second, when we consider only the two conditional-probability models, we can see that the additional information obtained from context in the prefix estimate gives a substantial improvement in this measure as compared to the trigram estimate.

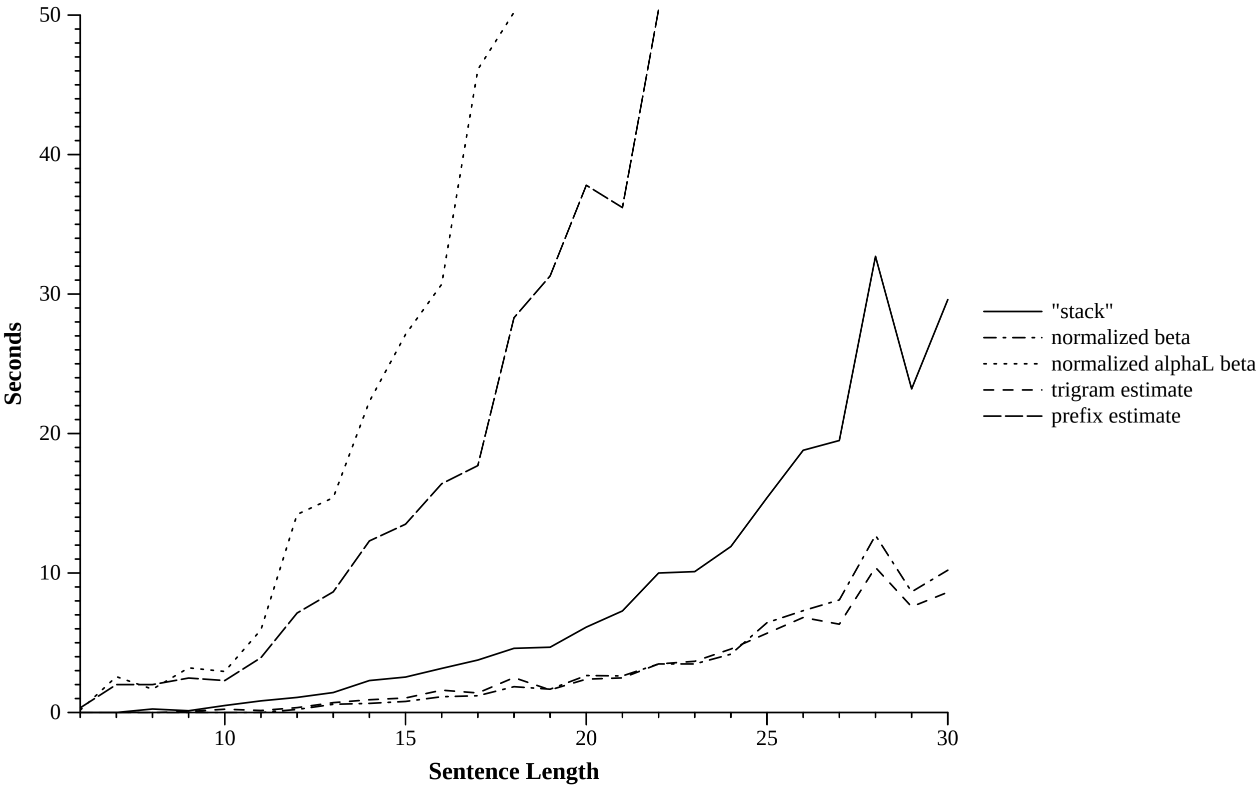

Figure 8: Average CPU Time for 95% of Probability Mass for AlphaL Beta Estimates

However, the CPU time needed to compute the α_L term exceeds the time saved by processing fewer edges. Note that using this estimate, the parser took over 26,000 seconds to get 95% of the probability mass, while the "stack" model can exhaustively parse the test data in less than 5,000 seconds. Figure 8 shows the average CPU time for each sentence length.

While chart parsing and calculations of β can be done in $O(n^3)$ time (see Appendix A), we have been unable to find an algorithm to compute the α_L terms faster than $O(n^5)$. When a constituent is removed from the agenda, it only affects the β values of its ancestors in the parse trees; however, α_L values are propagated to all of the constituent's siblings to the right and all of its descendants. Recomputing the α_L terms when a constituent is removed from the agenda can be done in $O(n^3)$ time, and since there are $O(n^2)$ possible constituents, the total time needed to compute the α_L terms in this manner is $O(n^5)$.

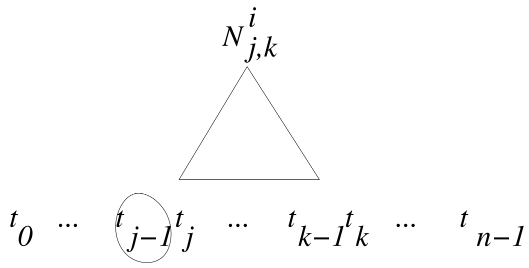

Figure 9: Left boundary context

5 Figures using boundary statistics

5.1 Left boundary trigram estimate

Although the α_L based models seem impractical, the edge-count and constituent-count statistics show that contextual information is useful. We can derive an estimate similar to the prefix estimate but containing a much simpler model of the context as follows:

$$\begin{aligned} p(N^i_{j,k}|t_{0,n}) &= \frac{p(N^i_{j,k}, t_{0,n})}{p(t_{0,n})} \\ &= \frac{p(t_{0,j}, t_{k,n}) p(N^i_{j,k}|t_{0,j}, t_{k,n}) p(t_{j,k}|N^i_{j,k}, t_{0,j}, t_{k,n})}{p(t_{0,j}, t_{k,n}) p(t_{j,k}|t_{0,j}, t_{k,n})}. \end{aligned}$$

Once again applying the usual independence assumption that given a nonterminal, the tag sequence it generates depends only on that nonterminal, we can rewrite the figure of merit as follows:

$$p(N^i_{j,k}|t_{0,n}) \approx \frac{p(N^i_{j,k}|t_{0,j}, t_{k,n}) \beta(N^i_{j,k})}{p(t_{j,k}|t_{0,j}, t_{k,n})}.$$

As usual, we use a trigram model for the tags, giving $p(t_{j,k}|t_{0,j}, t_{k,n}) \approx p(t_{j,k}|t_{j-2}, t_{j-1})$.

Now, we assume that $p(N^i_{j,k}|t_{0,j}, t_{k,n}) \approx p(N^i_{j,k}|t_{j-1})$, that is, that the probability of a nonterminal is dependent on the tag immediately before it in the sentence. (See Figure 9.) Then we have:

$$p(N^i_{j,k}|t_{0,n}) \approx \frac{p(N^i_{j,k}|t_{j-1}) \beta(N^i_{j,k})}{p(t_{j,k}|t_{j-2}, t_{j-1})}.$$

We can calculate $\beta(N^i_{j,k})$ and the tritag probabilities as usual. The $p(N^i_{j,k}|t_{j-1})$ probabilities are estimated from our training data by parsing the training data and counting the occurrences of the nonterminal and the tag weighted by their probability in the parse. (Further details are provided in Appendix B.)

We will refer to this figure as the **left boundary trigram estimate**.

5.2 Boundary trigram estimate

We can derive a similar estimate using context on both sides of the constituent as follows:

$$
\begin{aligned}
& p(N^i_{j,k}|t_{0,n}) \\
&= \frac{p(N^i_{j,k}, t_{0,n})}{p(t_{0,n})} \\
&= \frac{p(t_{0,j})p(N^i_{j,k}|t_{0,j})p(t_{j,k}|N^i_{j,k}, t_{0,j})p(t_k|t_{0,j}, N^i_{j,k}, t_{j,k})p(t_{k+1,n}|t_{0,j}, N^i_{j,k}, t_{j,k}, t_k)}{p(t_{0,j})p(t_{j,k}|t_{0,j})p(t_k|t_{0,k})p(t_{k+1,n}|t_{0,k+1})} \\
&= \frac{p(N^i_{j,k}|t_{0,j})p(t_{j,k}|N^i_{j,k}, t_{0,j})p(t_k|t_{0,j}, N^i_{j,k}, t_{j,k})p(t_{k+1,n}|t_{0,k+1}, N^i_{j,k})}{p(t_{j,k}|t_{0,j})p(t_k|t_{0,k})p(t_{k+1,n}|t_{0,k+1})}.
\end{aligned}
$$

Once again applying the usual independence assumption that given a nonterminal, the tag sequence it generates depends only on that nonterminal and also assuming that the probability of $t_{k+1,n}$ depends only on the previous tags, we can rewrite the figure of merit as follows:

$$p(N^i_{j,k}|t_{0,n}) \approx \frac{p(N^i_{j,k}|t_{0,j})\beta(N^i_{j,k})p(t_k|t_{0,k}, N^i_{j,k})}{p(t_{j,k+1}|t_{0,j})}.$$

Now we add some new independence assumptions. We assume that the probability of the nonterminal depends only on the immediately preceding tag, and that the probability of the tag immediately following the nonterminal depends only on the nonterminal, giving:

$$p(N^i_{j,k}|t_{0,n}) \approx \frac{p(N^i_{j,k}|t_{j-1})\beta(N^i_{j,k})p(t_k|N^i_{j,k})}{p(t_{j,k+1}|t_{0,j})}.$$

(See Figure 10.) As usual, we use a trigram model for the tags, giving $p(t_{j,k}|t_{0,j}, t_{k,n}) \approx p(t_{j,k}|t_{j-2}, t_{j-1})$. Then we have:

$$p(N^i_{j,k}|t_{0,n}) \approx \frac{p(N^i_{j,k}|t_{j-1})\beta(N^i_{j,k})p(t_k|N^i_{j,k})}{p(t_{j,k+1}|t_{j-2}, t_{j-1})}.$$

We can calculate $\beta(N^i_{j,k})$ and the tritag probabilities as usual. The $p(N^i_{j,k}|t_{j-1})$ and $p(t_k|N^i_{j,k})$ probabilities are estimated from our training data by parsing the training data and counting the occurrences of the nonterminal and the tag

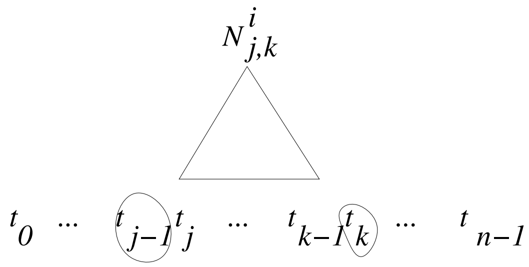

Figure 10: Boundary context

weighted by their probability in the parse.[2] Again, see Appendix B for details of how these estimates were obtained.

We will refer to this figure as the **boundary trigram estimate**.

5.3 Boundary statistics only

We also wished to examine if contextual information by itself is sufficient as a figure of merit. We can derive an estimate based only on easily-computable contextual information as follows:

$$
\begin{aligned}
& p(N^i_{j,k}|t_{0,n}) \\
&= \frac{p(N^i_{j,k}, t_{0,n})}{p(t_{0,n})} \\
&= \frac{p(t_{0,j})p(N^i_{j,k}|t_{0,j})p(t_{j,k}|N^i_{j,k}, t_{0,j})p(t_k|t_{0,j}, N^i_{j,k}, t_{j,k})p(t_{k+1,n}|t_{0,j}, N^i_{j,k}, t_{j,k}, t_k)}{p(t_{0,j})p(t_{j,k}|t_{0,j})p(t_k|t_{0,k})p(t_{k+1,n}|t_{0,k+1})} \\
&= \frac{p(N^i_{j,k}|t_{0,j})p(t_{j,k}|N^i_{j,k}, t_{0,j})p(t_k|t_{0,j}, N^i_{j,k}, t_{j,k})p(t_{k+1,n}|t_{0,k+1}, N^i_{j,k})}{p(t_{j,k}|t_{0,j})p(t_k|t_{0,k})p(t_{k+1,n}|t_{0,k+1})}.
\end{aligned}
$$

Most of the independence assumptions we make are the same as in the boundary trigram estimate. We assume that the probability of the nonterminal depends only on the previous tag, that the probability of the immediately following tag depends only on the nonterminal, and that the probability of the tags following that to depend only on the previous tags. However, we make one independence assumption that differs from all of our previous estimates. Rather than assuming the probability of the tags within the constituent depends on the nonterminal, giving an inside probability term, we assume that the probability of these tags depends only on the previous tags. Then we have

[2] Actually, in our implementation, the $p(t_k)$ in the denominator is included in the following-tag statistic, for which we actually use $\frac{p(t_k|N^i_{j,k})}{p(t_k)}$. Then at runtime we only use the trigram probabilities for $t_{0,k}$.

$$
\begin{aligned}
p(N^i_{j,k}|t_{0,n}) &\approx \frac{p(N^i_{j,k}|t_{0,j})p(t_{j,k}|t_{0,j})p(t_k|N^i_{j,k})p(t_{k+1,n}|t_{0,k+1})}{p(t_{j,k}|t_{0,j})p(t_k|t_{0,k})p(t_{k+1,n}|t_{0,k+1})} \\
&= \frac{p(N^i_{j,k}|t_{0,j})p(t_k|N^i_{j,k})}{p(t_k|t_{0,k})}.
\end{aligned}
$$

In the denominator, we take $p(t_k|t_{0,k}) \approx p(t_k)$, giving

$$
p(N^i_{j,k}|t_{0,n}) \approx \frac{p(N^i_{j,k}|t_{0,j})p(t_k|N^i_{j,k})}{p(t_k)}
$$

which is simply the product of the two boundary statistics described in the previous section.

We refer to this estimate as **boundary statistics only**.

5.4 Results

The results for the figures of merit introduced in the previous section according to the measurements given in Section 2.2 are as follows:

Figure of Merit	%E	%non-0 E	%popped	CPU time
boundary statistics only	53.2	50.8	59.6	2759
left boundary trigram estimate	22.1	18.4	39.6	1700
boundary trigram estimate	18.2	13.9	31.2	1111

Figure 11 shows a graph of %non-0 E for each sentence length for the boundary models and the trigram and prefix estimates.

This graph shows that the contextual information gained from using α_L in the prefix estimate is almost completely included in just the previous tag, as illustrated by the left boundary trigram estimate. Adding right contextual information in the boundary trigram estimate gives us the best performance on this measure of any of our figures of merit.

We can consider the left boundary trigram estimate to be an approximation of the prefix estimate, where the effect of the left context is approximated by the effect of the single tag to the left. Similarly, the boundary trigram estimate is an approximation to an estimate involving the full context, i.e., an estimate involving the outside probability α. However, the parser cannot compute the outside probability of a constituent during a parse, and so in order to use context on both sides of the constituent, we need to use something like our boundary statistics. Our results suggest that a single tag before or after the constituent can be used as a reasonable approximation to the full context on that side of the constituent.

Figure 12 shows the average CPU time for each sentence length.

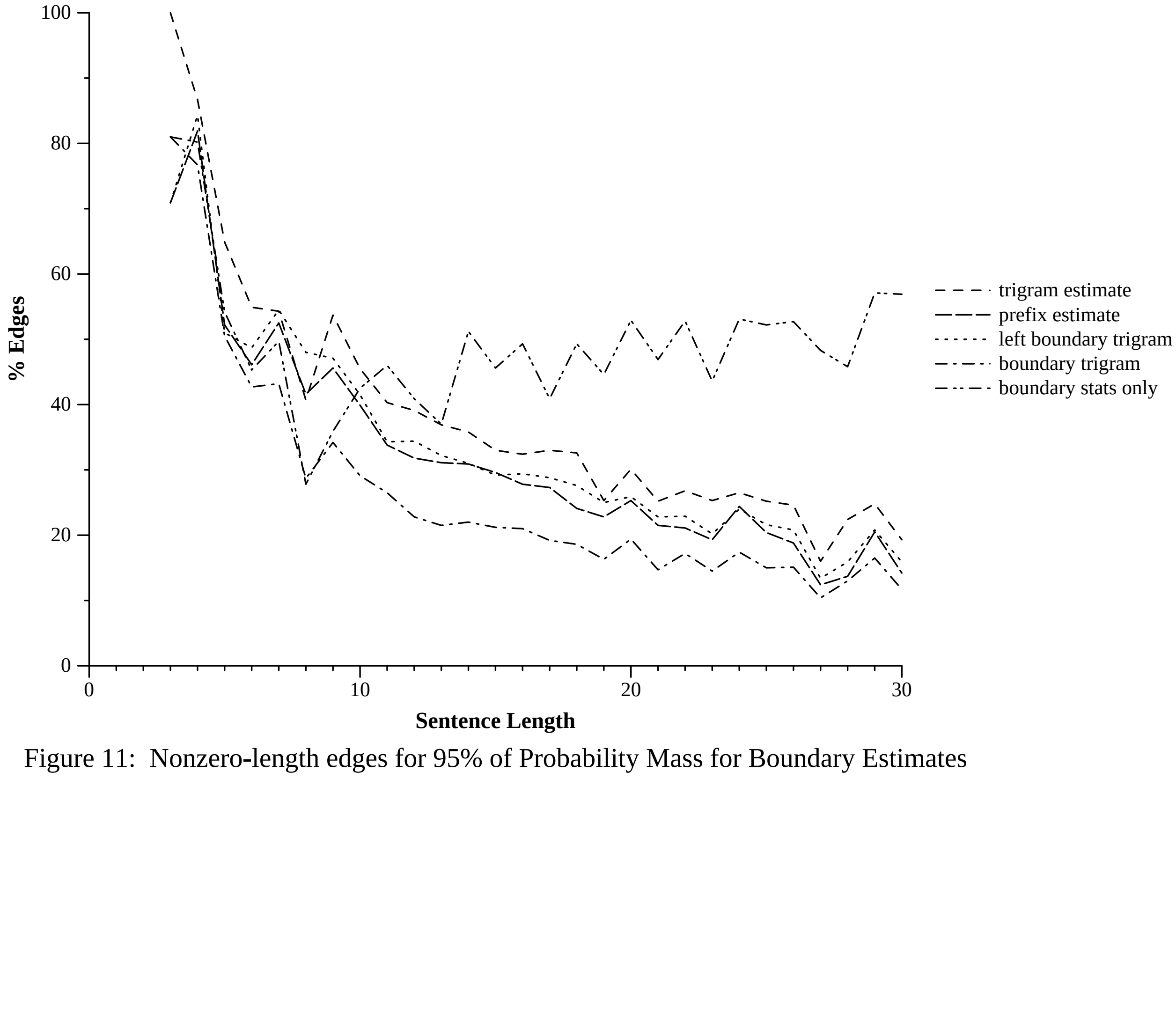

Figure 11: Nonzero-length edges for 95% of Probability Mass for Boundary Estimates

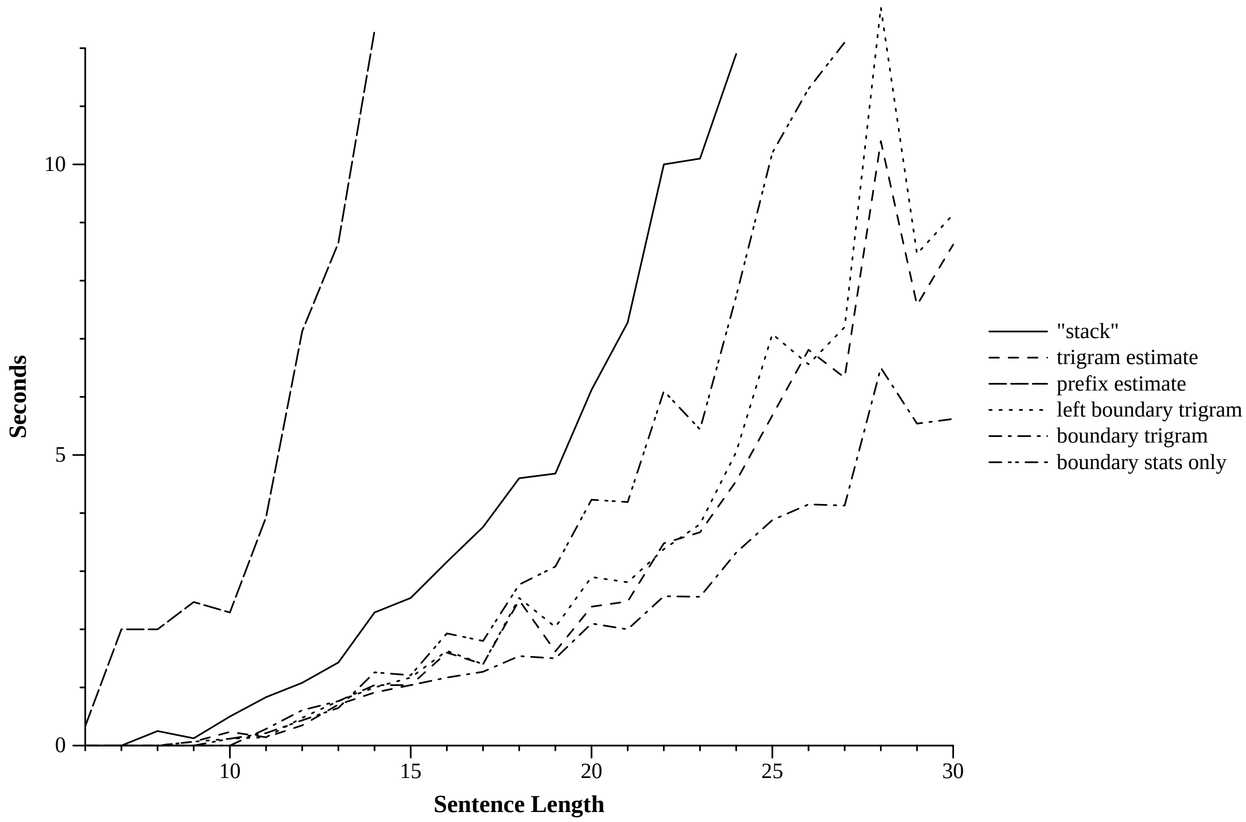

Figure 12: Average CPU Time for 95% of Probability Mass for Boundary Estimates

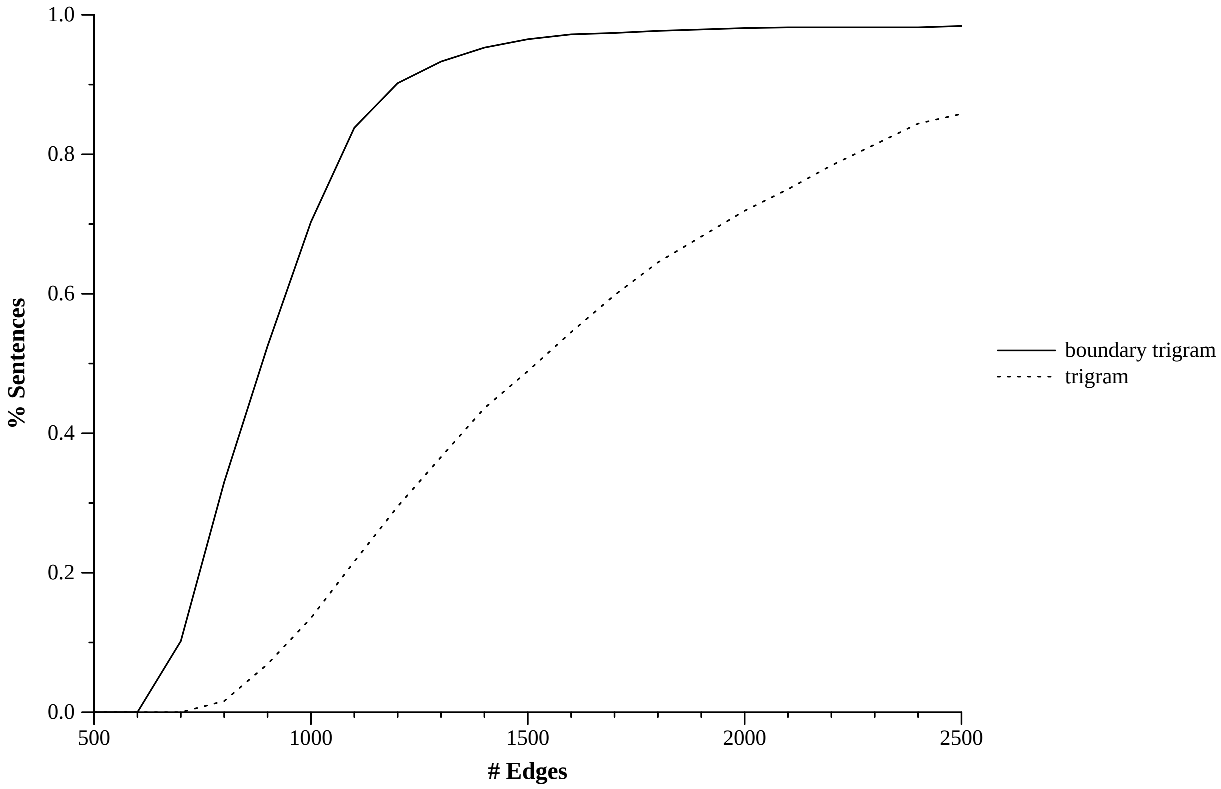

Figure 13: % 18-26 Word Sentences Finding a Parse in Fixed Number of Edges

Since the boundary trigram estimate has none of the overhead associated with the prefix estimate, it is the best performer in terms of CPU time as well. We can also see that using just the boundary statistics, which can be precomputed and require no extra processing during parsing, still results in a substantial improvement over the non-best-first "stack" model.

As another method of comparison between the two best-performing estimates, the contextually-dependent boundary trigram model and the context-independent trigram model, we compared the number of edges needed to find the first parse for average-length sentences. The average length of a sentence in our test data is about 22 words. Figure 13 shows the percentage of sentences of length 18 through 26 for which a parse could be found within 2500 edges. For this experiment, we used a separate test set from the Wall Street Journal corpus, containing approximately 570 sentences in the desired length range.

This measure also shows a real advantage of the boundary trigram estimate over the trigram estimate.

6 Results summary

The chart below summarizes the results obtained each figure of merit:

Figure of Merit	%E	%non-0 E	%popped	CPU time
"stack" model				4882
straight β	97.6	97.5	93.8	3966
normalized β	34.7	31.6	61.5	1631
trigram estimate	25.2	21.7	44.3	1547
normalized $\alpha_L \beta$	39.7	36.4	57.3	68660
prefix estimate	21.8	17.4	38.3	26520
boundary statistics only	53.2	50.8	59.6	2759
left boundary trigram estimate	22.1	18.4	39.6	1700
boundary trigram estimate	18.2	13.9	31.2	1111

7 Comparing figures of merit using a treebank grammar

7.1 Background

To verify that our results are not an artifact of the particular grammar we chose for testing, we also tested using a treebank grammar introduced in [6]. This grammar was trained in a straightforward way by reading the grammar directly (with minor modifications) from a portion of the Penn Treebank Wall Street Journal data comprised of about 300,000 words. The boundary statistics were counted directly from the training data as well. The treebank grammar is much larger and more ambiguous than our original grammar, containing about 16,000 rules and 78 terminal and nonterminal symbols, and it was impractical to parse sentences to exhaustion using our existing hardware, so the figures based on 95% of the probability mass could not be computed. We were able to use this grammar to compare the number of edges needed to find the first parse using the trigram and boundary trigram estimates.

7.2 Results

Figure 14 shows the percentage of sentences of length 18 through 26 for which a parse could be found within 20,000 edges. Again, we used a test set of approximately 570 sentences of the appropriate length from the Wall Street Journal corpus. Although the x-axis covers a much wider range than in Figure 13, the relationship between the two estimates is quite similar.

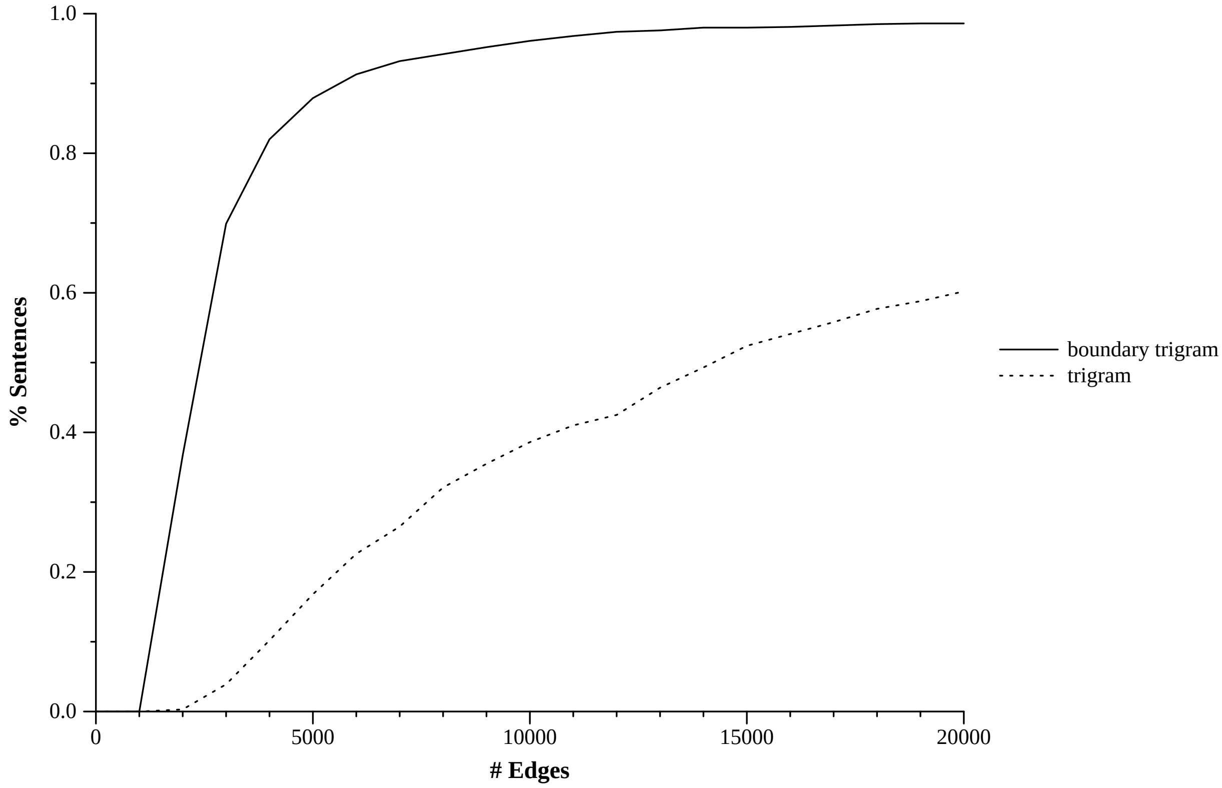

Figure 14: % 18-26 Word Sentences Finding a Parse in Fixed Number of Edges for Treebank Grammar

8 Previous work

In an earlier version of this paper [2], we presented the results for several of these models using our original grammar. The treebank grammar was introduced in [6], and the parser in that paper is a best-first parser using the boundary trigram figure of merit.

The literature shows many implementations of best-first parsing, but none of the previous work shares our goal of explicitly comparing figures of merit.

Bobrow [1] and Chitrao and Grishman [8] introduced statistical agenda-based parsing techniques. Chitrao and Grishman implemented a best-first probabilistic parser and noted the parser's tendency to prefer shorter constituents. They proposed a heuristic solution of penalizing shorter constituents by a fixed amount per word.

Miller and Fox [15] compare the performance of parsers using three different types of grammars, and show that a probabilistic context-free grammar using inside probability (unnormalized) as a figure of merit outperforms both a context-free grammar and a context-dependent grammar.

Kochman and Kupin [11] propose a figure of merit closely related to our prefix estimate. They do not actually incorporate this figure into a best-first parser.

Magerman and Marcus [12] use the geometric mean to compute a figure of merit that is independent of constituent length. Magerman and Weir [13] use a similar model with a different parsing algorithm.

9 Conclusions

We have presented and evaluated several figures of merit for best-first parsing. The best performer according to all of our measures was the parser using the boundary trigram estimate as a figure of merit, and this result holds for two different grammars. This figure has the additional advantage that it can be easily incorporated into existing best-first parsers using a figure of merit based on inside probability. (As mentioned earlier, the efficient online computation of β is described in Appendix A.) We strongly recommend this figure of merit as the basis for best-first statistical parsers.

The measurements presented here almost certainly underestimate the true benefits of this model. We restricted sentence length to a maximum of 30 words, in order to keep the number of edges in the exhaustive parse to a practical size; however, since the percentage of edges needed by the best-first parse decreases with increasing sentence length, we assume that the improvement would be even more dramatic for sentences longer than 30 words.

Acknowledgments

The authors are very grateful to Heidi Fox for obtaining the speed vs. accuracy data discussed in Appendix C. We also wish to thank the anonymous reviewers for their comments and suggestions.

References

[1] Robert J. Bobrow. Statistical agenda parsing. In *DARPA Speech and Language Workshop*, pages 222–224, 1990.

[2] Sharon Caraballo and Eugene Charniak. Figures of merit for best-first probabilistic chart parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 127–132, 1996.

[3] Glenn Carroll and Eugene Charniak. Learning probabilistic dependency grammars from labeled text. In *Working Notes*, Fall Symposium Series, pages 25–32. AAAI, 1992.

[4] Glenn Carroll and Eugene Charniak. Two experiments on learning probabilistic dependency grammars from corpora. In *Workshop Notes, Statistically-Based NLP Techniques*, pages 1–13. AAAI, 1992.

[5] Eugene Charniak. *Statistical Language Learning*. MIT Press, 1993.

[6] Eugene Charniak. Tree-bank grammars. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 1031–1036. AAAI, 1996.

[7] Eugene Charniak and Glenn Carroll. Context-sensitive statistics for improved grammatical language models. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 728–733, 1994.

[8] Mahesh V. Chitrao and Ralph Grishman. Statistical parsing of messages. In *DARPA Speech and Language Workshop*, pages 263–266, 1990.

[9] W. Nelson Francis and Henry Kučera. *Frequency Analysis of English Usage: Lexicon and Grammar*. Houghton Mifflin, 1982.

[10] Frederick Jelinek and John D. Lafferty. Computation of the probability of initial substring generation by stochastic context-free grammars. *Computational Linguistics*, 17:315–323, 1991.

[11] Fred Kochman and Joseph Kupin. Calculating the probability of a partial parse of a sentence. In *DARPA Speech and Language Workshop*, pages 237–240, 1991.

[12] David M. Magerman and Mitchell P. Marcus. Parsing the Voyager domain using Pearl. In *DARPA Speech and Language Workshop*, pages 231–236, 1991.

[13] David M. Magerman and Carl Weir. Efficiency, robustness and accuracy in Picky chart parsing. In *Proceedings of the 30th ACL Conference*, pages 40–47, 1992.

[14] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics*, 19:313–330, 1993.

[15] Scott Miller and Heidi Fox. Automatic grammar acquisition. In *Proceedings of the Human Language Technology Workshop*, pages 268–271, 1994.

Appendix A: Efficient online computation of β

We compute estimates of the inside probability β for each proposed constituent incrementally as new constituents are added to the chart. Initially, β is set to 1 for each terminal symbol, since our input is given as a stream of tags, which are our terminals. When a new proposed constituent is added to the agenda, its β estimate is set to its current inside probability according to the constituents already in the chart. However, as more constituents are added to the chart, we may find a new way to build up a proposed constituent, i.e., additional evidence for that proposed constituent, so we need to update the β for the proposed constituent (and also for affected constituents already in the chart, since these may in turn affect other proposed constituents).

These updates can be quite expensive in terms of CPU time. However, many of the updates are quite small, and do not affect the relative ordering of the proposed constituents on the agenda. Instead of propagating every change to β, then, we only want to propagate those changes that we expect to have an effect on this ordering. What we have done is to have each constituent store not only its β value, but also an increment. Increases to the inside probability are added not to β itself, but to this increment, until the increment exceeds some threshold. Experimentally we have found that we can avoid propagating increments until they exceed 1% of the current value of β with very little effect on the parser's selection of constituents from the agenda.

This thresholding on the propagation of β allows us to update the β values online while still keeping the performance of the parser as $O(n^3)$ empirically.

Appendix B: Estimation of boundary statistics

Our figures of merit incorporating boundary statistics use the figures $p(N^i_{j,k}|t_{j-1})$ to represent the effect of the left context and $\frac{p(t_k|N^i_{j,k})}{p(t_k)}$ to represent the effect of the right context. For our experiments with the first grammar, which was learned from training data taken from the Brown corpus, we estimated these statistics from the same training data.

First, we parsed the training data according to our grammar. (It was necessary to do this, rather than using the hand-annotated parses of the training data, because our grammar does not use the same set of nonterminals as the corpus; see [3], [4], [7] for details.) Since we use the tags as our input, the probability of a nonterminal appearing with a particular previous tab is the same as the probability of that nonterminal appearing in any sentence containing that tag.

We can then count the probability-weighted occurrences of a nonterminal given the previous tag as follows:

$$\begin{aligned} C(N^i_{j,k}, t_{j-1}) &= \sum_{w_{0,n} \text{ containing } t_{j-1}} p(N^i_{j,k}|w_{0,n}) \\ &= \frac{\alpha(N^i_{j,k})\beta(N^i_{j,k})}{p(w_{0,n})} \end{aligned}$$

That is, for each sentence which contains the previous tag t_{j-1}, we increment our count by the probability of the nonterminal $N^i_{j,k}$ occurring immediately following t_{j-1} in that sentence.

Since we have a complete parse, the inside and outside probabilities and the sentence probability can be easily computed. We can also obtain the count $C(t_{j-1})$ simply by counting the number of sentences in which that tag appears in position $j-1$. We then obtain the conditional probability for the left boundary statistic as follows:

$$p(N^i_{j,k}|t_{j-1}) = \frac{C(N^i_{j,k}, t_{j-1})}{C(t_{j-1})}$$

The right boundary statistic is computed in the corresponding way.

For the experiment using the treebank grammar, these statistics were obtained by counting directly from the Wall Street Journal treebank corpus, just as the grammar rules and trigram statistics were.

Appendix C: Speed vs. accuracy

As an additional verification of our results, we gathered data on speed versus accuracy. For this experiment, we used the probabilistic context-free grammar

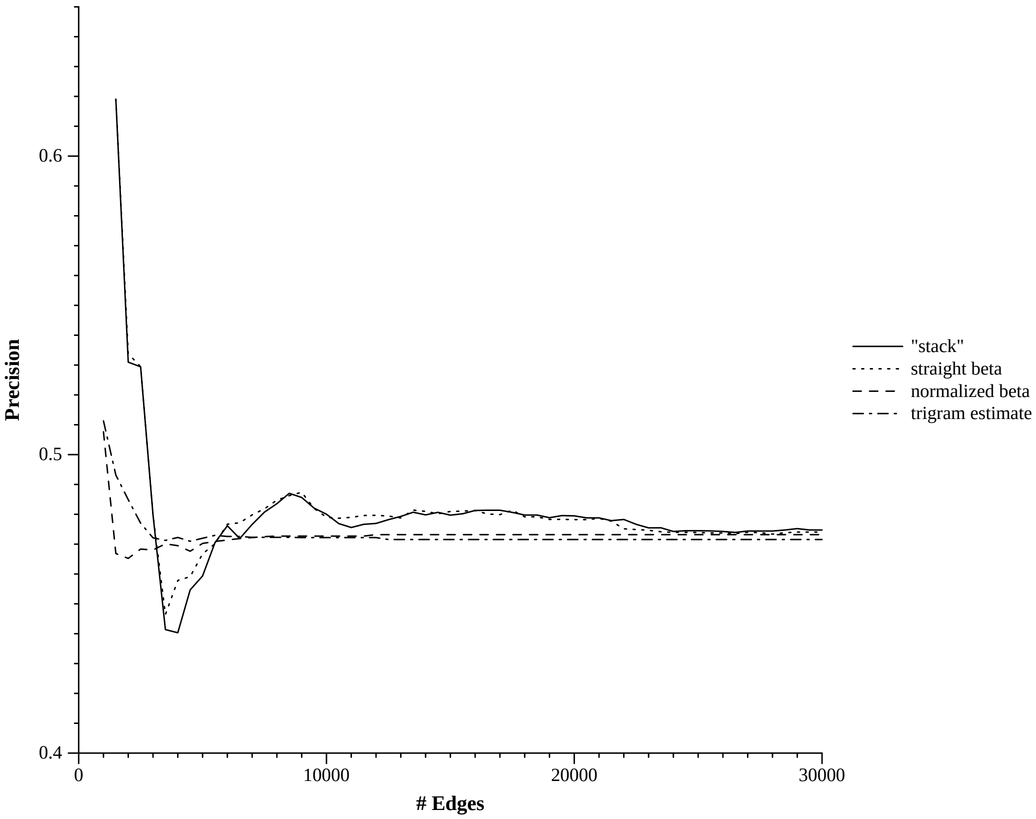

Figure C1: Precision of Best Parse Found in Fixed Number of Edges for Beta Estimates

learned from the Brown corpus and the average-length test sentences described in Section 5.4. For each figure of merit, we computed the average precision and recall of the best parse found as compared to the number of edges created. We computed unlabeled precision and recall only, since our grammar uses a different set of nonterminals from those used in the test data.

Precision is defined as the percentage of the constituents proposed by our parser which are actually correct according to the treebank. For each edge count, we measured the precision of the best parse of each sentence found within that number of edges. Figure C1 is a graph of the average precision for the β figures of merit from Section 3, plotted against edge counts.

The fluctuations at the low edge counts are due to the small amount of data at this level. At a low edge count, very few sentences have actually been parsed, and since these sentences tend to be short and simple, the parses are likely

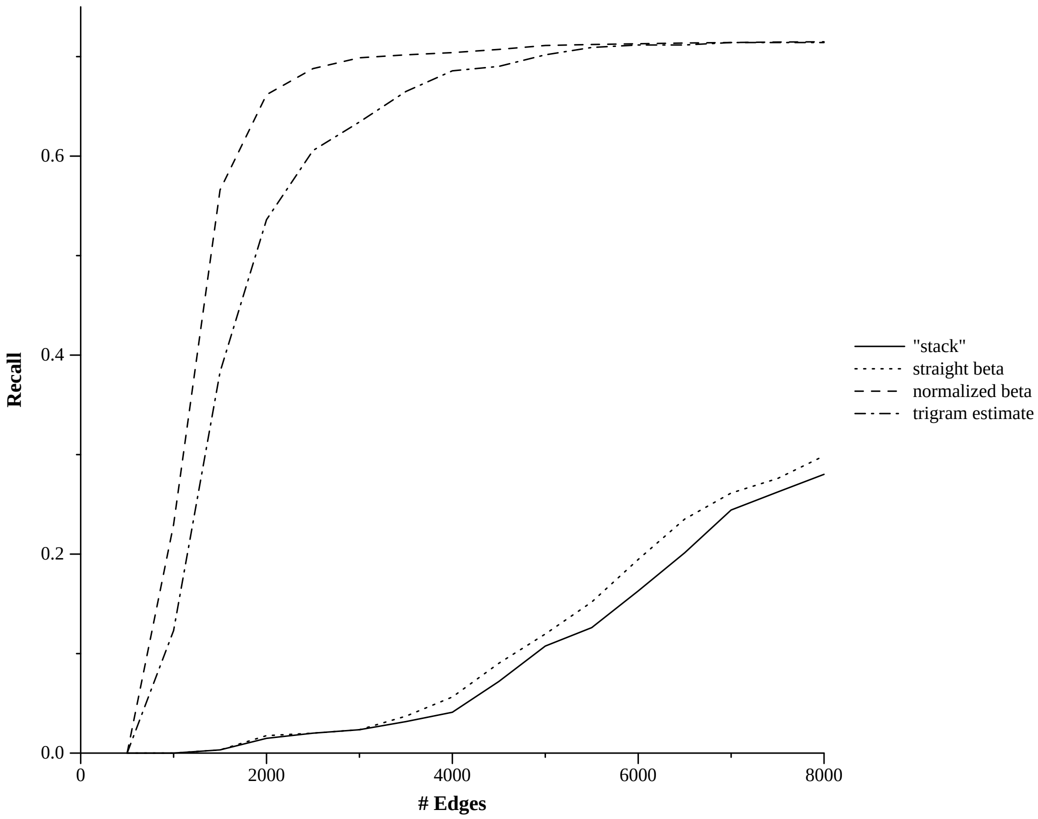

Figure C2: Recall of Best Parse Found in Fixed Number of Edges for Beta Estimates

to be correct. The sentences which could not be parsed do not contribute to the measurement of precision. As more sentences are parsed, precision settles at about 47%, the highest precision attainable by our particular test grammar, and remains there as edge counts increase. This level of precision is independent of the figure of merit used, so measurement of precision does not help evaluate our figures of merit.

A much more useful measure is recall. Recall is defined as the percentage of constituents in the treebank test data which are found by our parser. Again, we measured the recall of the best parse of each sentence found within each number of edges. Figure C2 shows the results for the figures of merit from Section 3.

Straight beta clearly shows little or no improvement over the "stack" parser using no figure of merit at all. The other figures of merit increase quickly to about 64%, the maximum recall attainable with our test grammar. The

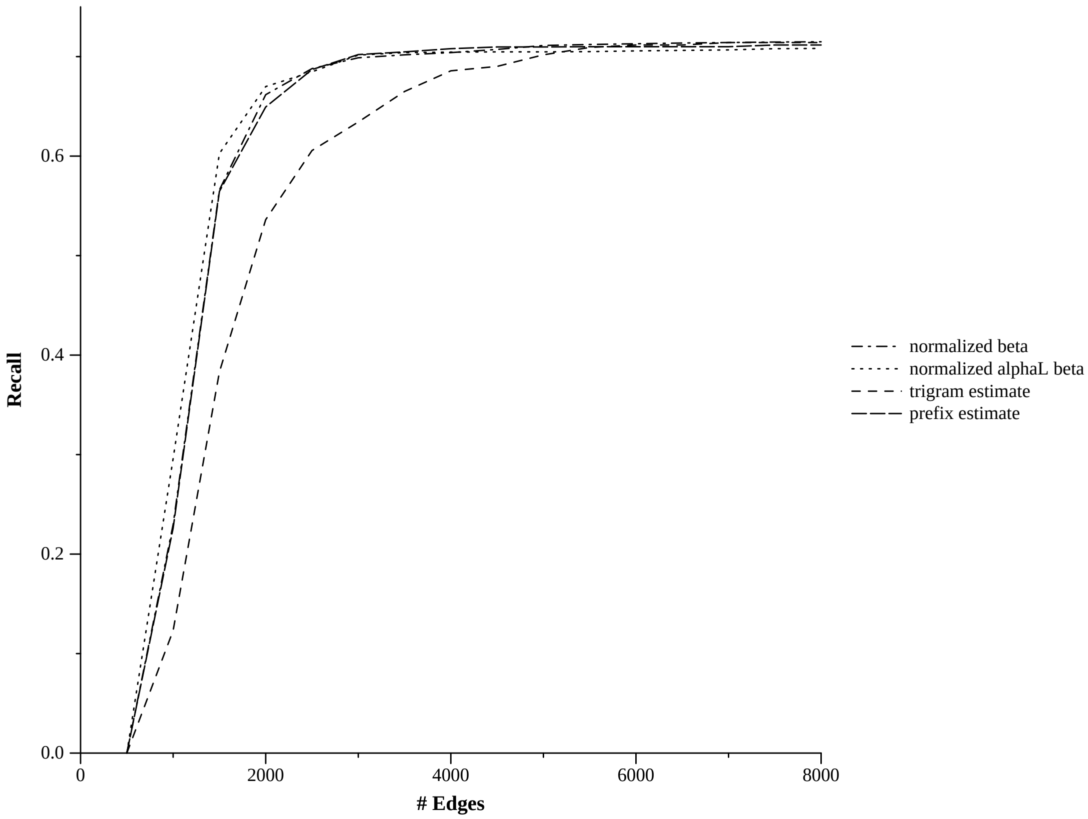

Figure C3: Recall of Best Parse Found in Fixed Number of Edges for AlphaL Beta Estimates

"stack" parser and the one using straight beta, on the other hand, do not reach this maximum level until about 50000 edges. We have no explanation for the relatively poor performance of the parser using the trigram estimate compared to the other best-first parsers, as shown here and in the following figures.

Figure C3 shows the recall values for the $\alpha_L\beta$ figures of merit from Section 4, and Figure C4 shows recall for the boundary figures of merit from Section 5. Since precision is not a useful measure, we have not included precision data for these figures of merit.

These data confirm that the parser using the boundary trigram figure of merit performs better than any of the others. Recall using this figure of merit is consistently higher than any of the others at low edge counts, and it reaches the maximum value in fewer than 2000 edges, with the nearest competitors approaching the maximum at about 3000 edges.

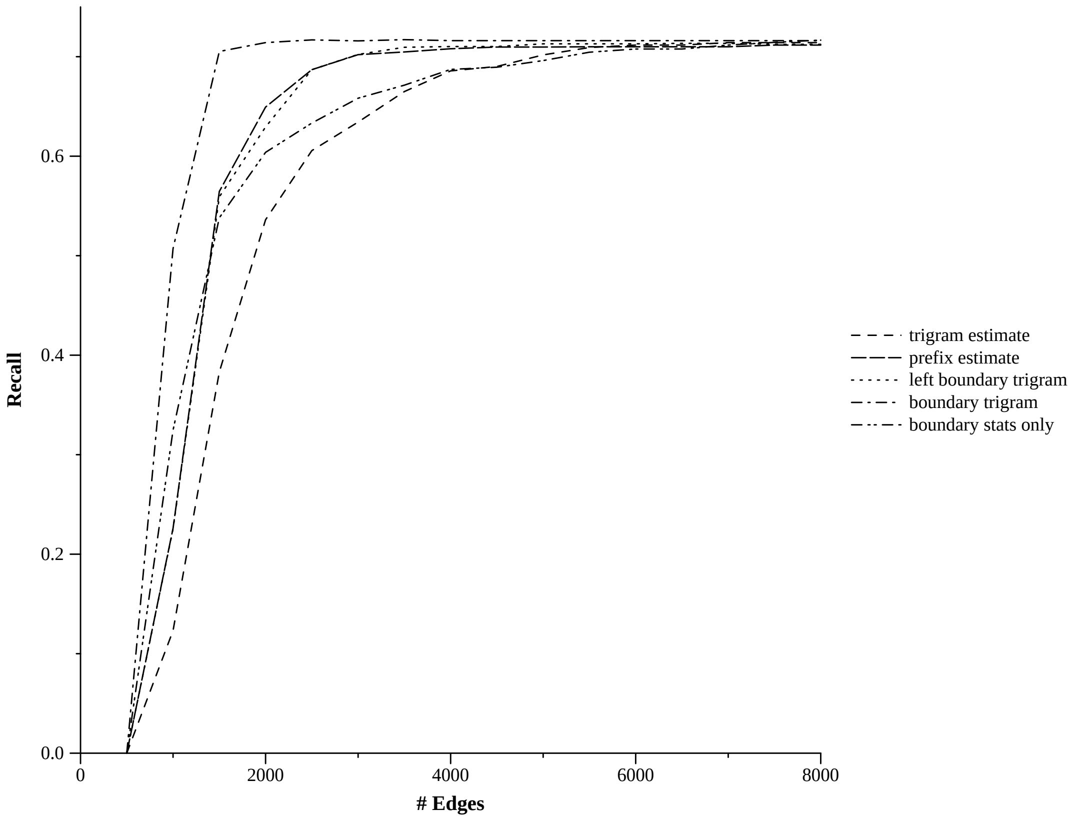

Figure C4: Recall of Best Parse Found in Fixed Number of Edges for Boundary Estimates