Automatic Compensation for Parser Figure-of-Merit Flaws*

Don Blaheta and Eugene Charniak
{dpb,ec}@cs.brown.edu
Department of Computer Science
Box 1910 / 115 Waterman St.—4th floor
Brown University
Providence, RI 02912

Abstract

Best-first chart parsing utilises a figure of merit (FOM) to efficiently guide a parse by first attending to those edges judged better. In the past it has usually been static; this paper will show that with some extra information, a parser can compensate for FOM flaws which otherwise slow it down. Our results are faster than the prior best by a factor of 2.5; and the speedup is won with no significant decrease in parser accuracy.

1 Introduction

Sentence parsing is a task which is traditionally rather computationally intensive. The best known practical methods are still roughly cubic in the length of the sentence—less than ideal when dealing with nontrivial sentences of 30 or 40 words in length, as frequently found in the Penn Wall Street Journal treebank corpus.

Fortunately, there is now a body of literature on methods to reduce parse time so that the exhaustive limit is never reached in practice.[1] For much of the work, the chosen vehicle is chart parsing. In this technique, the parser begins at the word or tag level and uses the rules of a context-free grammar to build larger and larger constituents. Completed constituents are stored in the cells of a chart according to their location and length. Incomplete constituents ("edges") are stored in an agenda. The exhaustion of the agenda definitively marks the completion of the parsing algorithm, but the parse needn't take that long; already in the early work on chart parsing, (Kay, 1970) suggests that by ordering the agenda one can find a parse without resorting to an exhaustive research. The introduction of statistical parsing brought with an obvious tactic for ranking the agenda: (Bobrow, 1990) and (Chitrao and Grishman, 1990) first used probabilistic context free grammars (PCFGs) to generate probabilities for use in a figure of merit (FOM). Later work introduced other FOMs formed from PCFG data (Kochman and Kupin, 1991); (Magerman and Marcus, 1991); and (Miller and Fox, 1994).

More recently, we have seen parse times lowered by several orders of magnitude. The (Caraballo and Charniak, 1998) article considers a number of different figures of merit for ordering the agenda, and ultimately recommends one that reduces the number of edges required for a full parse into the thousands. (Goldwater et al., 1998) (henceforth [Gold98]) introduces an edge-based technique, (instead of constituent-based), which drops the average edge count into the hundreds.

However, if we establish "perfection" as the minimum number of edges needed to generate the correct parse—47.5 edges on average in our corpus—we can hope for still more improvement. This paper looks at two new figures of merit, both of which take the [Gold98] figure (of "independent" merit) as a starting point in calculating a new figure

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

[1] An exhaustive parse always "overgenerates" because the grammar contains thousands of extremely rarely applied rules; these are (correctly) rejected even by the simplest parsers, eventually, but it would be better to avoid them entirely.

of merit for each edge, taking into account some additional information. Our work further lowers the average edge count, bringing it from the hundreds into the dozens.

2 Figure of independent merit

(Caraballo and Charniak, 1998) and [Gold98] use a figure which indicates the merit of a given constituent or edge, relative only to itself and its children but independent of the progress of the parse—we will call this the edge's *independent merit* (IM). The philosophical backing for this figure is that we would like to rank an edge based on the value

$$P(N^i_{j,k}|t_{0,n}) \ , \tag{1}$$

where $N^i_{j,k}$ represents an edge of type i (NP, S, etc.), which encompasses words j through $k-1$ of the sentence, and $t_{0,n}$ represents all n part-of-speech tags, from 0 to $n-1$. (As in the previous research, we simplify by looking at a tag stream, ignoring lexical information.) Given a few basic independence assumptions (Caraballo and Charniak, 1998), this value can be calculated as

$$P(N^i_{j,k}|t_{0,n}) \quad = \quad \frac{\beta(N^i_{j,k})\alpha(N^i_{j,k})}{P(t_{0,n})} \ , \tag{2}$$

with β and α representing the well-known "inside" and "outside" probability functions:

$$\begin{aligned} \beta(N^i_{j,k}) &= P(t_{j,k}|N^i_{j,k}) & (3) \\ \alpha(N^i_{j,k}) &= P(t_{0,j}, N^i_{j,k}, t_{k,n}) \ . & (4) \end{aligned}$$

Unfortunately, the outside probability is not calculable until after a parse is completed. Thus, the IM is an approximation; if we cannot calculate the full outside probability (the probability of this constituent occurring with all the other tags in the sentence), we can at least calculate the probability of this constituent occurring with the previous and subsequent tag. This approximation, as given in (Caraballo and Charniak, 1998), is

$$\frac{P(N^i_{j,k}|t_{j-1})\beta(N^i_{j,k})P(t_k|N^i_{j,k})}{P(t_{j,k}|t_{j-1})P(t_k|t_{k-1})} \ . \tag{5}$$

Of the five values required, $P(N^i_{j,k}|t_j)$, $P(t_k|t_{k-1})$, and $P(t_k|N^i_{j,k})$ can be observed directly from the training data; the inside probability is estimated using the most probable parse for $N^i_{j,k}$, and the tag sequence probability is estimated using a bitag approximation.

Two different probability distributions are used in this estimate, and the PCFG probabilities in the numerator tend to be a bit lower than the bitag probabilities in the denominator; this is more of a factor in larger constituents, so the figure tends to favour the smaller ones. To adjust the distributions to counteract this effect, we will use a normalisation constant η as in [Gold98]. Effectively, the inside probability β is multiplied by η^{k-j}, preventing the discrepancy and hence the preference for shorter edges. In this paper we will use $\eta = 1.3$ throughout; this is the factor by which the two distributions differ, and was also empirically shown to be the best tradeoff between number of popped edges and accuracy (in [Gold98]).

3 Finding FOM flaws

Clearly, any improvement to be had would need to come through eliminating the incorrect edges before they are popped from the agenda—that is, improving the figure of merit. We observed that the FOMs used tended to cause the algorithm to spend too much time in one area of a sentence, generating multiple parses for the same substring, before it would generate even one parse for another area. The reason for that is that the figures of independent merit are frequently good as relative measures for ranking different parses of the same section of the sentence, but not so good as absolute measures for ranking parses of different substrings.

For instance, if the word "there" as an NP in "there's a hole in the bucket" had a low probability, it would tend to hold up the parsing of a sentence; since the bi-tag probability of "there" occurring at the beginning of a sentence is very high, the denominator of the IM would overbalance the numerator. (Note that this is a contrived example—the actual problem cases are more obscure.) Of course, a different figure of independent merit might have different characteristics, but with many of them there will be cases where the figure is flawed, causing a single, vital edge to remain on the agenda while the parser 'thrashes' around in other parts of the sentence with higher IM values.

We could characterise this observation as follows:

Postulate 1 *The longer an edge stays in the agenda without any competitors, the more likely it is to be correct (even if it has a low figure of independent merit).*

A better figure, then, would take into account whether a given piece of text had already been parsed or not. We took two approaches to finding such a figure.

4 Compensating for flaws

4.1 Experiment 1: Table lookup

In one approach to the problem, we tried to start our program with no extra information and train it statistically to counter the problem mentioned in the previous section. There are four values mentioned in Postulate 1: correctness, time (amount of work done), number of competitors, and figure of independent merit. We defined them as follows:

Correctness. The obvious definition is that an edge $N^i_{j,k}$ is correct if a constituent $N^i_{j,k}$ appears in the parse given in the treebank. There is an unobvious but unfortunate consequence of choosing this definition, however; in many cases (especially with larger constituents), the "correct" rule appears just once in the entire corpus, and is thus considered too unlikely to be chosen by the parser as correct. If the "correct" parse were never achieved, we wouldn't have any statistic at all as to the likelihood of the first, second, or third competitor being better than the others. If we define "correct" for the purpose of statistics-gathering as "in the MAP parse", the problem is diminished. Both definitions were tried for gathering statistics, though of course only the first was used for measuring accuracy of output parses.

Work. Here, the most logical measure for amount of work done is the number of edges popped off the agenda. We use it both because it is conveniently processor-independent and because it offers us a tangible measure of perfection (47.5 edges—the average number of edges in the correct parse of a sentence).

Competitorship. At the most basic level, the competitors of a given edge $N^i_{j,k}$ would be all those edges $N^x_{m,n}$ such that $m \leq j$ and $n \geq k$. Initially we only considered an edge a 'competitor' if it met this definition and were already in the chart; later we tried considering an edge to be a competitor if it had a higher independent merit, no matter whether it be in the agenda or the chart. We also tried a hybrid of the two.

Merit. The independent merit of an edge is defined in section 2. Unlike earlier work, which used what we call "Independent Merit" as the FOM for parsing, we use this figure as just one of many sources of information about a given edge.

Given our postulate, the ideal figure of merit would be

$$P(\mathit{correct} \mid W, C, \mathit{IM}) \,. \qquad (6)$$

We can save information about this probability for each edge in every parse; but to be useful in a statistical model, the IM must first be discretised, and all three prior statistics need to be grouped, to avoid sparse data problems. We bucketed all three logarithmically, with bases 4, 2, and 10, respectively. This gives us the following approximation:

$$P(\mathit{correct} \mid \lfloor \log_4 W \rfloor, \lfloor \log_2 C \rfloor, \lfloor \log_{10} \mathit{IM} \rfloor) \,. \qquad (7)$$

To somewhat counteract the effect of discretising the IM figure, each time we needed

$$FOM = P\left(correct \mid \lfloor \log_4 W \rfloor, \lfloor \log_2 C \rfloor, \lfloor \log_{10} IM \rfloor\right)\left(\lceil \log_{10} IM \rceil - \log_{10} IM\right) + P\left(correct \mid \lfloor \log_4 W \rfloor, \lfloor \log_2 C \rfloor, \lceil \log_{10} IM \rceil\right)\left(\log_{10} IM - \lfloor \log_{10} IM \rfloor\right) \quad (8)$$

to calculate a figure of merit, we looked up the table entry on either side of the IM and interpolated. Thus the actual value used as a figure of merit was that given in equation (8).

Each trial consisted of a training run and a testing run. The training runs consisted of using a grammar induced on treebank sections 2–21 to run the edge-based best-first algorithm (with the IM alone as figure of merit) on section 24, collecting the statistics along the way. It seems relatively obvious that each edge should be counted when it is created. But our postulate involves edges which have stayed on the agenda for a long time without accumulating competitors; thus we wanted to update our counts when an edge happened to get more competitors, and as time passed. Whenever the number of edges popped crossed into a new logarithmic bucket (i.e. whenever it passed a power of four), we re-counted every edge in the agenda in that new bucket. In addition, when the number of competitors of a given edge passed a bucket boundary (power of two), that edge would be re-counted. In this manner, we had a count of exactly how many edges—correct or not—had a given IM and a given number of competitors at a given point in the parse.

Already at this stage we found strong evidence for our postulate. We were paying particular attention to those edges with a low IM and zero competitors, because those were the edges that were causing problems when the parser ignored them. When, considering this subset of edges, we looked at a graph of the percentage of edges in the agenda which were correct, we saw an increase of orders of magnitude as work increased—see Figure 1.

For the testing runs, then, we used as figure of merit the value in expression 8. Aside from that change, we used the same edge-based best-first parsing algorithm as before. The test runs were all made on treebank sec-

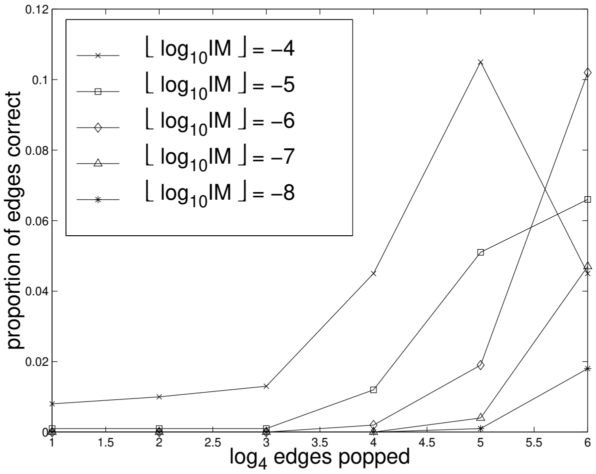

Figure 1: *Zero competitors, low IM—Proportion of agenda edges correct vs. work*

tion 22, with all sentences longer than 40 words thrown out; thus our results can be directly compared to those in the previous work.

We made several trials, using different definitions of 'correct' and 'competitor', as described above. Some performed much better than others, as seen in Table 1, which gives our results, both in terms of accuracy and speed, as compared to the best previous result, given in [Gold98]. The trial descriptions refer back to the multiple definitions given for 'correct' and 'competitor' at the beginning of this section. While our best speed improvement (48.6% of the previous minimum) was achieved with the first run, it is associated with a significant loss in accuracy. Our best results overall, listed in the last row of the table, let us cut the edge count by almost half while reducing labelled precision/recall by only 0.24%.

4.2 Experiment 2: Demeriting

We hoped, however, that we might be able to find a way to simplify the algorithm such that it would be easier to implement and/or

Table 1: Performance of various statistical schemata

Trial description	Labelled Precision	Labelled Recall	Change in LP/LR avg.	Edges popped[2]	Percent of std.
[Gold98] standard	75.814%	73.334%		229.73	
Correct, Chart competitors	74.982%	72.920%	-.623%	111.59	48.6%
Correct, higher-merit competitors	75.588%	73.190%	-.185%	135.23	58.9%
Correct, Chart or higher-merit	75.433%	73.152%	-.282%	128.94	56.1%
MAP, higher-merit competitors	75.365%	73.220%	-.239%	120.47	52.4%

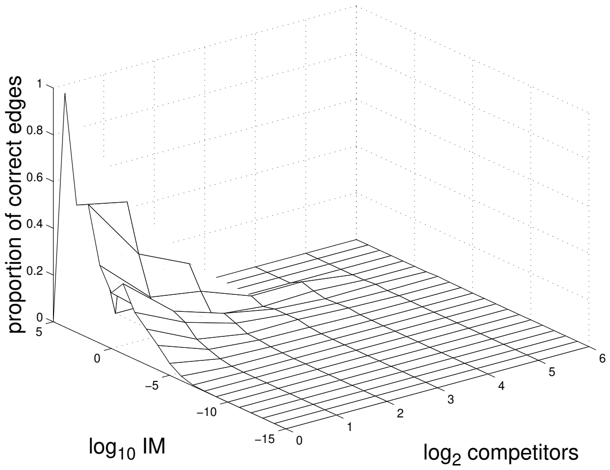

Figure 2: *Stats at 64–255 edges popped*

faster to run, without sacrificing accuracy. To that end, we looked over the data, viewing it as (among other things) a series of "planes" seen by setting the amount of work constant (see Figure 2). Viewed like this, the original algorithm behaves like a scan line, parallel to the competitor axis, scanning for the one edge with the highest figure of (independent) merit. However, one look at figure 2 dramatically confirms our postulate—that an edge with zero competitors can have an IM orders of magnitude lower than an edge with many competitors, and still be more likely to be correct. Effectively, then, under the table lookup algorithm, the scan line is not parallel to the competitor axis, but rather angled so that the low-IM low-competitor items pass the scan before the high-IM high-competitor items. This can be simulated by multiplying each edge's independent merit by a *de*meriting factor δ per competitor (thus a total of δ^C). Its exact value would determine the steepness of the scan line.

Each trial consisted of one run, an edge-based best-first parse of treebank section 22 (with sentences longer than 40 words thrown out, as before), using the new figure of merit:

$$\delta^C \left(\frac{\eta^{k-j} P(N^i_{j,k}|t_{j-1}) \beta(N^i_{j,k}) P(t_k|N^i_{j,k})}{P(t_{j,k}|t_{j-1}) P(t_k|t_{k-1})} \right) . \quad (9)$$

This idea works extremely well. It is, predictably, easier to implement; somewhat surprisingly, though, it actually performs better than the method it approximates. When $\delta = .7$, for instance, the accuracy loss is only .28%, comparable to the table lookup result, but the number of edges popped drops to just 91.23, or 39.7% of the prior result found in [Gold98]. Using other demeriting factors gives similarly dramatic decreases in edge count, with varying effects on accuracy—see Figures 3 and 4.

It is not immediately clear as to why demeriting improves performance so dramatically over the table lookup method. One possibility is that the statistical method runs into too many sparse data problems around the fringe of the data set—were we able to use a larger data set, we might see the statistics approach the curve defined by the demeriting. Another is that the bucketing is too coarse, although the interpolation along

---

[2] Previous work has shown that the parser performs better if it runs slightly past the first parse; so for every run referenced in this paper, the parser was allowed to run to first parse plus a tenth. All reported final counts for popped edges are thus 1.1 times the count at first parse.

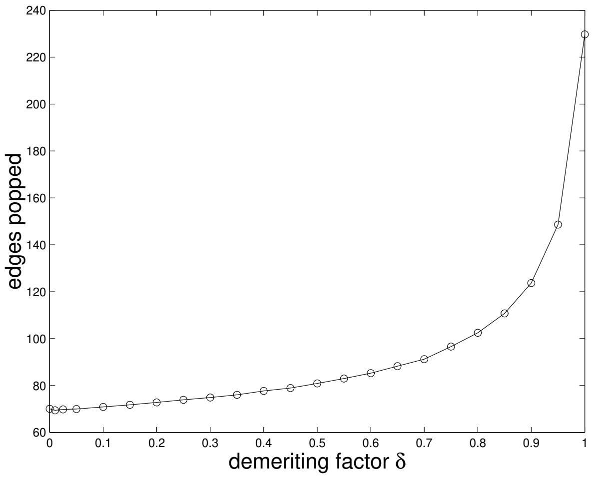

Figure 3: *Edges popped vs.* δ

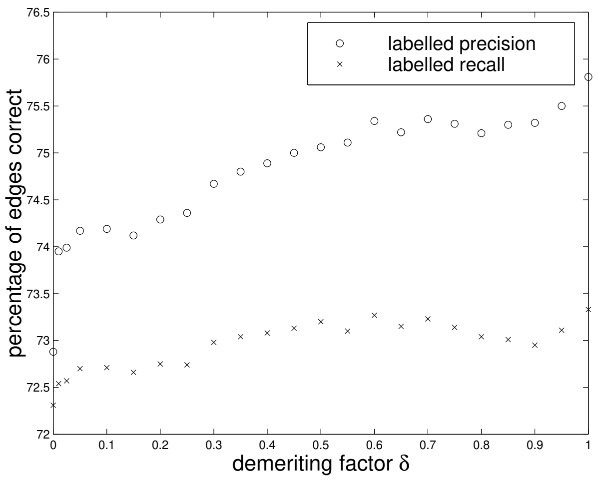

Figure 4: *Precision and recall vs.* δ

the independent merit axis would seem to mitigate that problem.

5 Conclusion

In the prior work, we see the average edge cost of a chart parse reduced from 170,000 or so down to 229.7. This paper gives a simple modification to the [Gold98] algorithm that further reduces this count to just over 90 edges, less than two times the perfect minimum number of edges. In addition to speeding up tag-stream parsers, it seems reasonable to assume that the demeriting system would work in other classes of parsers such as the lexicalised model of (Charniak, 1997)—as long as the parsing technique has some sort of demeritable ranking system, or at least some way of paying less attention to already-filled positions, the kernel of the system should be applicable. Furthermore, because of its ease of implementation, we strongly recommend the demeriting system to those working with best-first parsing.

References

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

Sharon Caraballo and Eugene Charniak. 1998. New figures of merit for best-first probabilistic chart parsing. *Computational Linguistics*, 24(2):275–298, June.

Eugene Charniak. 1997. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 598–603, Menlo Park. AAAI Press/MIT Press.

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

Sharon Goldwater, Eugene Charniak, and Mark Johnson. 1998. Best-first edge-based chart parsing. In *6th Annual Workshop for Very Large Corpora*, pages 127–133.

Martin Kay. 1970. Algorithm schemata and data structures in syntactic processing. In Barbara J. Grosz, Karen Sparck Jones, and Bonne Lynn Weber, editors, *Readings in Natural Language Processing*, pages 35–70. Morgan Kaufmann, Los Altos, CA.

Fred Kochman and Joseph Kupin. 1991. Calculating the probability of a partial parse of a sentence. In *DARPA Speech and Language Workshop*, pages 273–240.

David M. Magerman and Mitchell P. Marcus. 1991. Parsing the voyager domain using pearl. In *DARPA Speech and Language Workshop*, pages 231–236.

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