Probabilistic Prediction and Picky Chart Parsing*

David M. Magerman
Stanford University
magerman@cs.stanford.edu

Carl Weir
Paramax Systems
weir@prc.unisys.com

ABSTRACT

This paper describes Picky, a probabilistic agenda-based chart parsing algorithm which uses a technique called *probabilistic prediction* to predict which grammar rules are likely to lead to an acceptable parse of the input. In tests on randomly selected test data, Picky generates fewer edges on average than other CKY-like algorithms, while achieving 89% first parse accuracy and also enabling the parser to process sentences with false starts and other minor disfluencies. Further, sentences which are parsed completely by the probabilistic prediction technique have a 97% first parse accuracy.

1. Introduction

Two important concerns in natural language parsing which encourage the use of probabilistic analysis are efficiency and accuracy. An accurate parser which has a domain or language model so detailed that it takes hours to process a single sentence, while perhaps interesting, is no more useful than a simple instantaneous parser which is always wrong. Probabilistic modelling of the grammar of a language has been proposed as a potential solution to the accuracy problem, disambiguating grammatical parses generated by an ambiguous grammar. However, little attention has been paid to the repercussions of probabilistic parsing on the computational complexity and average-case performance of existing parsing algorithms. Effective probabilistic models of grammar which take into account contextual information (e.g. [10] [2]) cannot take advantage of the $O(n^3)$ behavior of CKY-like parsing algorithms. If they are to use existing grammatical formalisms, these models *must* use algorithms that are worst-case exponential.

When natural language parsers are incorporated into natural language understanding systems, another significant issue arises: robustness. As a component of a language processing system, a parser's task is to analyze correctly all inputs which can be understood by the system, not just those which are precisely grammatical. Or, one might say, the grammar of natural language includes fragments, run-ons, split infinitives, and other disfluencies which would receive red marks on a high school English paper. At the same time, meaningless sequences of words and other uninterpretable inputs should *not* be analyzed as though they are acceptable. Robust processing of natural language is an ideal application of probabilistic methods, since probability theory provides a well-behaved measure of expectation within a given language.

This paper proposes an agenda-based probabilistic chart parsing algorithm which is both robust and efficient. The algorithm, Picky[1], is considered robust because it will potentially generate all constituents produced by a pure bottom-up parser and rank these constituents by likelihood. The efficiency of the algorithm is achieved through a technique called *probabilistic prediction,* which helps the algorithm avoid worst-case behavior. Probabilistic prediction is a trainable technique for modelling where edges are likely to occur in the chart-parsing process.[2] Once predicted edges are added to the chart using probabilistic prediction, they are processed in a style similar to agenda-based chart parsing algorithms. By limiting the edges in the chart to those which are predicted by this model, the parser can process a sentence while generating only the most likely constituents given the input.

The Picky parsing algorithm is divided into three phases, where the goal of each phase is to minimize the set of rule predictions in the chart to only those necessary to generate an analysis of the input sentence. When a phase completes without producing an analysis of the input, the next phase expands the set of rules which it can use and applies these new rules to the chart from the previous phase. The proposed algorithm is still exponential in the worst-case, but only exhibits worst-case behavior on sentences which are completely outside the domain of the training material (i.e. contain multiple occurrences of grammatical structures rarely seen or unseen in training). In this work, the efficiency of various

*Special thanks to Jerry Hobbs and Bob Moore at SRI for providing access to their computers, and to Salim Roukos, Peter Brown, and Vincent and Steven Della Pietra at IBM for their instructive lessons on probabilistic modelling of natural language.

[1] Pearl ≡ probabilistic Earley-style parser (P-Earl). Picky ≡ probabilistic CKY-like parser (P-CKY).

[2] Some familiarity with chart parsing terminology is assumed in this paper. For terminological definitions, see [7], [8], [9], or [14].

algorithms and effectiveness of models is determined by a comparison of the number of rule predicts, rule advancing operations (the basic operation in chart parsing), and complete constituents detected by the parser.

The results of experiments using this parsing algorithm are quite promising. On a corpus of 300 randomly selected test sentences, Picky parses these sentences with 89% first parse accuracy, and up to 92% accuracy within the first three parses. Further, sentences which are parsed completely by the probabilistic prediction technique, in phases I and II, have a 97% first parse accuracy. The algorithm is extremely efficient, with less than a 1.6:1 ratio of constituents recognized to constituents in the final parse for sentences parsed by phases I and II. The performance decreases for sentence outside the training corpus that are parsed in phase III.

This paper will present the Picky parsing algorithm, describing the both the original features of the parser and those adapted from previous work. Then, along with accuracy and efficiency results, the paper will report an analysis of the interaction between the phases of the parsing algorithm and the probabilistic models of parsing and prediction.

2. Probabilistic Models

The probabilistic models used in the implementation of Picky are independent of the algorithm. To facilitate the comparison between the performance of Picky and its predecessor, Pearl, the probabilistic model implemented for Picky is similar to Pearl's scoring model, the context-free grammar with context-sensitive probability (CFG with CSP) model. This probabilistic model estimates the probability of each parse T given the words in the sentence S, $\mathcal{P}(T|S)$, by assuming that each non-terminal and its immediate children are dependent on the non-terminal's siblings and parent and on the part-of-speech trigram centered at the beginning of that rule:

$$\mathcal{P}(T|S) \simeq \prod_{A \in T} \mathcal{P}(A \rightarrow \alpha | C \rightarrow \beta A \gamma,\ a_0 a_1 a_2) \quad (1)$$

where C is the non-terminal node which immediately dominates A, a_1 is the part-of-speech associated with the leftmost word of constituent A, and a_0 and a_2 are the parts-of-speech of the words to the left and to the right of a_1, respectively. See Magerman and Marcus 1991 [10] for a more detailed description of the CFG with CSP model.

3. The Parsing Algorithm

A probabilistic language model, such as the aforementioned CFG with CSP model, provides a metric for evaluating the likelihood of a parse tree. However, while it may *suggest* a method for evaluating partial parse trees, a language model alone does not dictate the search strategy for determining the most likely analysis of an input. Since exhaustive search of the space of parse trees produced by a natural language grammar is generally not feasible, a parsing model can best take advantage of a probabilistic language model by incorporating it into a parser which probabilistically models the parsing process. Picky attempts to model the chart parsing process for context-free grammars using probabilistic prediction.

Picky parses sentences in three phases: *covered left-corner* phase (I), *covered bidirectional* phase (II), and *tree completion* phase (III). Each phase uses a different method for proposing edges to be introduced to the parse chart. The first phase, covered left-corner, uses probabilistic prediction based on the left-corner word of the left-most daughter of a constituent to propose edges. The covered bidirectional phase also uses probabilistic prediction, but it allows prediction to occur from the left-corner word of any daughter of a constituent, and parses that constituent outward (bidirectionally) from that daughter. These phases are referred to as "covered" because, during these phases, the parsing mechanism proposes only edges that have non-zero probability according to the prediction model, i.e. that have been covered by the training process. The final phase, tree completion, is essentially an exhaustive search of all interpretations of the input according to the grammar. However, the search proceeds in best-first order, according to the measures provided by the language model. This phase is used only when the probabilistic prediction model fails to propose the edges necessary to complete a parse of the sentence.

The following sections will present and motivate the prediction techniques used by the algorithm, and will then describe how they are implemented in each phase.

3.1. Probabilistic Prediction

Probabilistic prediction is a general method for using probabilistic information extracted from a parsed corpus to estimate the likelihood that predicting an edge at a certain point in the chart will lead to a correct analysis of the sentence. The Picky algorithm is not dependent on the specific probabilistic prediction model used. The model used in the implementation, which is similar to the probabilistic language model, will be described.[3]

The prediction model used in the implementation of

[3] It is not necessary for the prediction model to be the same as the language model used to evaluate complete analyses. However, it is helpful if this is the case, so that the probability estimates of incomplete edges will be consistent with the probability estimates of completed constituents.

Picky estimates the probability that an edge proposed at a point in the chart will lead to a correct parse to be:

$$\mathcal{P}(A \rightarrow \alpha B \beta | a_0 a_1 a_2), \quad (2)$$

where a_1 is the part-of-speech of the left-corner word of B, a_0 is the part-of-speech of the word to the left of a_1, and a_2 is the part-of-speech of the word to the right of a_1.

To illustrate how this model is used, consider the sentence

The cow raced past the barn. (3)

The word "cow" in the word sequence "the cow raced" predicts **NP → det n,** but not **NP → det n PP,** since PP is unlikely to generate a verb, based on training material.[4] Assuming the prediction model is well trained, it *will* propose the interpretation of "raced" as the beginning of a participial phrase modifying "the cow," as in

The cow raced past the barn mooed. (4)

However, the interpretation of "raced" as a past participle will receive a low probability estimate relative to the verb interpretation, since the prediction model only considers local context.

The process of probabilistic prediction is analogous to that of a human parser recognizing predictive lexical items or sequences in a sentence and using these hints to restrict the search for the correct analysis of the sentence. For instance, a sentence beginning with a *wh*-word and auxiliary inversion is very likely to be a question, and trying to interpret it as an assertion is wasteful. If a verb is generally ditransitive, one should look for two objects to that verb instead of one or none. Using probabilistic prediction, sentences whose interpretations are highly predictable based on the trained parsing model can be analyzed with little wasted effort, generating sometimes no more than ten spurious constituents for sentences which contain between 30 and 40 constituents! Also, in some of these cases every predicted rule results in a completed constituent, indicating that the model made *no* incorrect predictions and was led astray only by genuine ambiguities in parts of the sentence.

3.2. Exhaustive Prediction

When probabilistic prediction fails to generate the edges necessary to complete a parse of the sentence, exhaustive prediction uses the edges which have been generated in earlier phases to predict new edges which might combine with them to produce a complete parse. Exhaustive prediction is a combination of two existing types of prediction, "over-the-top" prediction [9] and top-down filtering.

Over-the-top prediction is applied to complete edges. A completed edge $A \rightarrow \alpha$ will predict all edges of the form $B \rightarrow \beta A \gamma$.[5]

Top-down filtering is used to predict edges in order to complete incomplete edges. An edge of the form $A \rightarrow \alpha B_0 B_1 B_2 \beta$, where a B_1 has been recognized, will predict edges of the form $B_0 \rightarrow \gamma$ before B_1 and edges of the form $B_2 \rightarrow \delta$ after B_1.

3.3. Bidirectional Parsing

The only difference between phases I and II is that phase II allows bidirectional parsing. Bidirectional parsing is a technique for initiating the parsing of a constituent from any point in that constituent. Chart parsing algorithms generally process constituents from left-to-right. For instance, given a grammar rule

$$A \rightarrow B_1 B_2 \cdots B_n, \quad (5)$$

a parser generally would attempt to recognize a B_1, then search for a B_2 following it, and so on. Bidirectional parsing recognizes an A by looking for any B_i. Once a B_i has been parsed, a bidirectional parser looks for a B_{i-1} to the left of the B_i, a B_{i+1} to the right, and so on.

Bidirectional parsing is generally an inefficient technique, since it allows duplicate edges to be introduced into the chart. As an example, consider a context-free rule NP → DET N, and assume that there is a determiner followed by a noun in the sentence being parsed. Using bidirectional parsing, this NP rule can be predicted both by the determiner and by the noun. The edge predicted by the determiner will look to the right for a noun, find one, and introduce a new edge consisting of a completed NP. The edge predicted by the noun will look to the left for a determiner, find one, and also introduce a new edge consisting of a completed NP. Both of these NPs represent identical parse trees, and are thus redundant. If the algorithm permits both edges to be inserted into the chart, then an edge XP → α NP β will be advanced by both NPs, creating two copies of every XP edge. These duplicate XP edges can themselves be used in other rules, and so on.

[4] Throughout this discussion, we will describe the prediction process using words as the predictors of edges. In the implementation, due to sparse data concerns, only parts-of-speech are used to predict edges. Given more robust estimation techniques, a probabilistic prediction model conditioned on word sequences is likely to perform as well or better.

[5] In the implementation of Picky, over-the-top prediction for $A \rightarrow \alpha$ will only predict edges of the form $B \rightarrow A\gamma$. This limitation on over-the-top prediction is due to the expensive bookkeeping involved in bidirectional parsing. See the section on bidirectional parsing for more details.

To avoid this propagation of redundant edges, the parser must ensure that no duplicate edges are introduced into the chart. $\mathcal{P}$icky does this simply by verifying every time an edge is added that the edge is not already in the chart.

Although eliminating redundant edges prevents excessive inefficiency, bidirectional parsing may still perform more work than traditional left-to-right parsing. In the previous example, three edges are introduced into the chart to parse the NP $\rightarrow$ DET N edge. A left-to-right parser would only introduce two edges, one when the determiner is recognized, and another when the noun is recognized.

The benefit of bidirectional parsing can be seen when probabilistic prediction is introduced into the parser. Frequently, the syntactic structure of a constituent is not determined by its left-corner word. For instance, in the sequence V NP PP, the prepositional phrase PP can modify either the noun phrase NP or the entire verb phrase V NP. These two interpretations require different VP rules to be predicted, but the decision about which rule to use depends on more than just the verb. The correct rule may best be predicted by knowing the preposition used in the PP. Using probabilistic prediction, the decision is made by pursuing the rule which has the highest probability according to the prediction model. This rule is then parsed bidirectionally. If this rule is in fact the correct rule to analyze the constituent, then no other predictions will be made for that constituent, and there will be no more edges produced than in left-to-right parsing. Thus, the only case where bidirectional parsing is less efficient than left-to-right parsing is when the prediction model fails to capture the elements of context of the sentence which determine its correct interpretation.

3.4. The Three Phases of $\mathcal{P}$icky

Covered Left-Corner The first phase uses probabilistic prediction based on the part-of-speech sequences from the input sentence to predict all grammar rules which have a non-zero probability of being dominated by that trigram (based on the training corpus), i.e.

$$\mathcal{P}(A \rightarrow B\delta | a_0 a_1 a_2) > 0 \qquad (6)$$

where a_1 is the part-of-speech of the left-corner word of B. In this phase, the only exception to the probabilistic prediction is that any rule which can immediately dominate the preterminal category of any word in the sentence is also predicted, regardless of its probability. This type of prediction is referred to as *exhaustive prediction.* All of the predicted rules are processed using a standard best-first agenda processing algorithm, where the highest scoring edge in the chart is advanced.

Covered Bidirectional If an S spanning the entire word string is not recognized by the end of the first phase, the covered bidirectional phase continues the parsing process. Using the chart generated by the first phase, rules are predicted not only by the trigram centered at the left-corner word of the rule, but by the trigram centered at the left-corner word of any of the children of that rule, i.e.

$$\mathcal{P}(A \rightarrow \alpha B\delta | b_0 b_1 b_2) > 0. \qquad (7)$$

where b_1 is the part-of-speech associated with the leftmost word of constituent B. This phase introduces incomplete theories into the chart which need to be expanded to the left and to the right, as described in the bidirectional parsing section above.

Tree Completion If the bidirectional processing fails to produce a successful parse, then it is assumed that there is some part of the input sentence which is not covered well by the training material. In the final phase, exhaustive prediction is performed on all complete theories which were introduced in the previous phases but which are not predicted by the trigrams beneath them (i.e. $\mathcal{P}(\text{rule} \mid \text{trigram}) = 0$).

In this phase, edges are only predicted by their left-corner word. As mentioned previously, bidirectional parsing can be inefficient when the prediction model is inaccurate. Since all edges which the prediction model assigns non-zero probability have already been predicted, the model can no longer provide any information for future predictions. Thus, bidirectional parsing in this phase is very likely to be inefficient. Edges already in the chart will be parsed bidirectionally, since they were predicted by the model, but all new edges will be predicted by the left-corner word only.

Since it is already known that the prediction model will assign a zero probability to these rules, these predictions are instead scored based on the number of words spanned by the subtree which predicted them. Thus, this phase processes longer theories by introducing rules which can advance them. Each new theory which is proposed by the parsing process is exhaustively predicted for, using the length-based scoring model.

The final phase is used only when a sentence is so far outside of the scope of the training material that none of the previous phases are able to process it. This phase of the algorithm exhibits the worst-case exponential behavior that is found in chart parsers which do not use node packing. Since the probabilistic model is no longer useful in this phase, the parser is forced to propose an enormous number of theories. The expectation (or hope) is that one of the theories which spans most of the sen-

tence will be completed by this final process. Depending on the size of the grammar used, it may be unfeasible to allow the parser to exhaust all possible predicts before deciding an input is ungrammatical. The question of when the parser should give up is an empirical issue which will not be explored here.

Post-processing: Partial Parsing Once the final phase has exhausted all predictions made by the grammar, or more likely, once the probability of all edges in the chart falls below a certain threshold, Picky determines the sentence to be ungrammatical. However, since the chart produced by Picky contains all recognized constituents, sorted by probability, the chart can be used to extract partial parses. As implemented, Picky prints out the most probable completed S constituent.

4. Results of Experiments

The Picky parser was tested on 3 sets of 100 sentences which were held out from the rest of the corpus during training. The training corpus consisted of 982 sentences which were parsed using the same grammar that Picky used. The training and test corpora are samples from the MIT's Voyager direction-finding system.[6] Our experiments explored the accuracy, efficiency, and robustness of the Picky algorithm.

However, we do not anticipate a significant improvement in accuracy, since the two parsers use similar language models. On the other hand, Picky should outperform Pearl in terms of robustness and efficiency.

4.1. Robustness

Since our test sets did not contain many ungrammatical sentences, it was difficult to analyze Picky's robustness. It is undeniable that Picky will produce a fuller chart than will Pearl, making partial parsing of ungrammatical sentences possible. We leave it to future experiments to explore empirically the effectiveness of semantic interpretation using Picky's probabilistic well-formed substring table.

One interesting example did occur in one test set. The sentence "How do I how do I get to MIT?" is a ungrammatical but interpretable sentence which begins with a restart. Pearl would have generated no analysis for the latter part of the sentence and the corresponding sections of the chart would be empty. Using bidirectional probabilistic prediction, Picky produced a correct partial interpretation of the last 6 words of the sentence, "how do I get to MIT?" One sentence does not make for conclusive evidence, but it represents the type of improvements which are expected from the Picky algorithm.

[6] Special thanks to Victor Zue at MIT for the use of the speech data from MIT's Voyager system.

4.2. Accuracy

Phase	No.	Accuracy
I + II	238	97%
III	62	60%
Overall	300	89.3%

Figure 1: Picky's parsing accuracy, categorized by the phase which the parser reached in processing the test sentences.

As we expected, Picky's parsing accuracy compares favorably to Pearl's performance. As shown in Figure 1, Picky parsed the test sentences with an 89.3% accuracy rate. This is a slight improvement over Pearl's 87.5% accuracy rate reported in [10].

But note the accuracy results for phases I and II. These phases include sentences which are parsed successfully by the probabilistic prediction mechanism. Almost 80% of the test sentences fall into this category, and 97% of these sentences are parsed correctly. This result is very significant because it provides a reliable measure of the confidence the parser has in it's interpretation. If incorrect interpretations are worse than no interpretation at all, a natural language system might consider only parses which are generated in phases I and II. This would limit coverage, but would allow the system to have a high degree of confidence in the parser output.

4.3. Efficiency

Phase	Parse	Predicts	Completes
I	37	57	58
II	41	89	98
III	44	315	430
Overall	38	111	135

Figure 2: Average number of edges generated by Picky, categorized by the phase which the parser reached in processing the test sentences.

The effectiveness of the prediction model also leads to increased efficiency. Figure 2 shows the average number of edges predicted and completed by sentences, again partitioned by phase of parse completion. Also included in the table is the average number of constituents in the "correct" parse.

A measure of the efficiency provided by the probabilistic prediction mechanism is the parser's *prediction ratio,* the ratio of edges predicted to edges necessary for a correct

parse. A perfect prediction ratio is 1:1, i.e. every edge predicted is used in the eventual parse. However, since there is ambiguity in the input sentences, a 1:1 prediction ratio is not likely to be achieved. Picky's prediction ratio is less than 3:1, and its ratio of predicted edges to completed edges is nearly 1:1. Thus, although the prediction ratio is not perfect, on average for every edge that is predicted one completed constituent results.

Note that the prediction ratio is much lower in phase I (1.5:1) and phase II (2.2:1) than in phase III (7:1). This is due to the accuracy of the probabilistic prediction model used in the first two phases, and the deficiencies of the heuristic model used in final phase. Further efficiency can be gained either by limiting the amount of search which is performed in phase III before a sentence is deemed ungrammatical or by improving the heuristic prediction model.

Since Picky has the power of a pure bottom-up parser, it would be useful to compare it's performance and efficiency to that of a probabilistic bottom-up parser. However, an implementation of a probabilistic bottom-up parser using the same grammar produces on average over 1000 constituents for each sentence, generating over 15,000 edges without generating a parse at all! This supports our claim that exhaustive CKY-like parsing algorithms are not feasible when probabilistic models are applied to them.

5. Conclusions

One of the goals of the development of the Picky algorithm is to demonstrate the need to model the parsing process as well as modelling language. The exponential behavior of statistical methods applied to standard parsing algorithms limits the types of stochastic grammars which can feasibly be used in natural language understanding systems. As the statistical models of natural language become richer and more expensive to compute, it is vital that we have efficient probabilistic parsing algorithms which avoid spurious generation of constituents and partial constituents, since each edge produced by the parsing process must be evaluated by the statistical language model. Picky successfully employs probabilistic prediction to minimize the number of constituents to which the language model must be applied, making complicated language models which use fine-grained statistics more feasible for natural language applications.

References

1. Bobrow, R. J. 1991. Statistical Agenda Parsing. In Proceedings of the February 1991 DARPA Speech and Natural Language Workshop. Asilomar, California.
2. Chitrao, M. and Grishman, R. 1990. Statistical Parsing of Messages. In Proceedings of the June 1990 DARPA Speech and Natural Language Workshop. Hidden Valley, Pennsylvania.
3. Church, K. 1988. A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text. In Proceedings of the Second Conference on Applied Natural Language Processing. Austin, Texas.
4. Earley, J. 1970. An Efficient Context-Free Parsing Algorithm. *Communications of the ACM* Vol. 13, No. 2, pp. 94-102.
5. Gale, W. A. and Church, K. 1990. Poor Estimates of Context are Worse than None. In Proceedings of the June 1990 DARPA Speech and Natural Language Workshop. Hidden Valley, Pennsylvania.
6. Jelinek, F. 1985. Self-organizing Language Modeling for Speech Recognition. IBM Report.
7. Kasami, T. 1965. An Efficient Recognition and Syntax Algorithm for Context-Free Languages. Scientific Report AFCRL-65-758, Air Force Cambridge Research Laboratory. Bedford, Massachusetts.
8. Kay, M. 1980. Algorithm Schemata and Data Structures in Syntactic Processing. *CSL-80-12*, October 1980.
9. Kimball, J. 1973. Principles of Surface Structure Parsing in Natural Language. *Cognition*, 2.15-47.
10. Magerman, D. M. and Marcus, M. P. 1991. Pearl: A Probabilistic Chart Parser. In Proceedings of the European ACL Conference, March 1991. Berlin, Germany.
11. Moore, R. and Dowding, J. 1991. Efficient Bottom-Up Parsing. In Proceedings of the February 1991 DARPA Speech and Natural Language Workshop. Asilomar, California. Berlin, Germany.
12. Sharman, R. A., Jelinek, F., and Mercer, R. 1990. Generating a Grammar for Statistical Training. In Proceedings of the June 1990 DARPA Speech and Natural Language Workshop. Hidden Valley, Pennsylvania.
13. Seneff, Stephanie 1989. TINA. In Proceedings of the August 1989 International Workshop in Parsing Technologies. Pittsburgh, Pennsylvania.
14. Younger, D. H. 1967. Recognition and Parsing of Context-Free Languages in Time n^3. *Information and Control* Vol. 10, No. 2, pp. 189-208.