Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 501–504
ISBN 978-83-60810-27-9
ISSN 1896-7094

Effective natural language parsing with probabilistic grammars

Paweł Skórzewski
Adam Mickiewicz University
Faculty of Mathematics and Computer Science
ul. Umultowska 87, 61-614 Poznań, Poland
Email: pawel@st.amu.edu.pl

***Abstract*—This paper presents an example of application of a PCFG parsing algorithm based on the A* search procedure in a machine translation system. We modified the existing CYK-based parser used in the machine translation system Translatica and applied the A* parsing algorithm in order to improve the performance of the parser.**

I. Introduction

PROBABILISTIC context-free grammars are a useful tool for modeling natural languages. One of the methods commonly used for parsing PCFGs is chart parsing. In general, the time complexity of chart parsing algorithms is worst-case cubic. In practice, for large grammars and long sentences, even cubic time can be insufficient.

The process of parsing PCFGs can be viewed as a process of finding a path in a certain weighted multigraph. This means that graph search algorithms can be used for parsing. The parse which has the highest probability from all the possible parses of the sentence is called the *Viterbi parse*. Finding a Viterbi parse of a given sentence can be realized by algorithms that find the shortest path in a graph. The graph algorithms are used to accelerate the chart parsing of PCFGs.

One of the methods of speeding up the parsing is the best-first strategy, described in [1] and [2]. The order of removing edges from agenda is determined by a figure of merit. If the figure of merit is properly chosen, the number of edges processed during the parsing can be significantly reduced.

Another method is a beam-search algorithm, described in [5] and [6]. In this method, only a limited number of best parses is tracked at any time. The fewer parses are tracked, the shorter time is required to complete the algorithm.

Both best-first parsing and beam-search parsing are greedy algorithms. Unfortunately, neither of them guarantees the finding of the actual Viterbi parse of a given sentence.

The A* algorithm is used to find the shortest path between two given vertices of a graph, assuming a heuristic function is specified. The heuristic function is defined on the vertices of the graph and estimates the distance between the given vertex and the target vertex. The algorithm tries to minimize the sum of the distance covered so far and the value of the heuristic.

Klein and Manning in [3] presented the algorithm that uses the A* procedure to parse the probabilistic context-free grammar. The authors showed that if a proper heuristic function is applied, the A* parsing algorithm can significantly reduce the time required to find the Viterbi parse of a given sentence.

II. The A* parsing algorithm

The A* parser is a kind of a chart parser. The basic data structure the parser operates on is called an edge. The edge is a triple $X[i,j]$, where X is a symbol of the grammar and (i,j) is the span of this symbol in the parse tree of the string $w = w_1 \dots w_n$, ie. X dominates the substring $w_{i+1} \dots w_j$. During the algorithm, a score is attached to each edge. The edges waiting to be processed are stored in the structure called agenda. The edges whose best parses have already been found are stored in a chart.

In the beginning, the edges representing the words of the parsed sentence are put into the agenda. Then, while the agenda is not empty, the following procedure (called a turn of the parser) is run: The edge with the highest priority is removed from the agenda. Then if the removed edge can be combined with some edges already in the chart then new edges are formed and inserted into the agenda. In other words, let $e = X[i,j]$ denote the removed edge. If there is an edge $Y[k,i]$ in the chart and a rule $Z \to YX$ in the set of productions, the new edge $Z[k,j]$ is inserted into the agenda. Similarly, if there is an edge $Y[j,k]$ in the chart and a rule $Z \to XY$ in the rule set, the new edge $Z[i,k]$ is formed etc. The rules of the grammar are used to combine the current edge with the symbols in the chart to create new edges and insert the newly formed edges into the agenda. The newly formed edges are put into the agenda and the process repeats until the agenda is empty.

Let $G = (V,T,R,S,P)$ be a probabilistic context-free grammar and $w = w_1 \dots w_n \in T^+$ a parsed sentence. Following [3], we will define the *Viterbi inside score* $\beta(e) = \beta_{G,w}(e)$ of an edge $e = X[i,j]$ as the logarithm of the probability of a best inside parse of e, ie. the maximum of the log-probabilities of the derivation $X \Rightarrow^* w_{i+1} \dots w_j$. We will denote by $b(e)$ the estimate of $\beta(e)$ which represent the log-probability of the best inside parse of e calculated so far. At the beginning of the algorithm $b(e) = -\infty$ and never decreases. When e is removed from agenda, $b(e) = \beta(e)$.

The *Viterbi outside score* $\alpha(e) = \alpha_{G,w}(e)$ of an edge $e = X[i,j]$ is defined as the maximum of the log-probabilities

of the derivation $S \Rightarrow^* w_1 \dots w_i X w_{j+1} \dots w_n$ (or the log-probability of the best outside parse of e). We will also estimate the value of $\alpha(e)$ by $a(e)$. We say that the estimation a is *admissible* if $a(e) \geq \alpha(e)$.

Provided that the value of $\beta + a$ never increases, we can use $\beta + a$ to prioritize edges in agenda. The β corresponds to the actual distance from the start vertex and a plays the role of the heuristic for the A* algorithm.

Klein and Manning in [3] present a series of different heuristic functions, based both on context summary estimates and grammar projection estimates.

The context summary estimates are based on the knowledge about the neighborhood of the current symbol. The value of the context summary estimate $a(X[i,j])$ is the maximum of all log-probabilities of the derivations $S \Rightarrow^* uXv$, where $uandv$ satisfies some given condition. There is a wide range of possible estimates, between the NULL estimate (when we have no information about the context), equal constantly 0, and the TRUE esimate (when we have the full information), equal the exact value of β.

The grammar projection estimates use the exact context but the reduced grammar. A probabilistic context-free grammar $G = (V, T, R, S, P)$ can be projected to some weighted context-free grammar $G' = (V', T', R', S', P')$ by a given function $\pi\colon V \cup T \to N$, where N is an arbitrary set of symbols, in the following way:

- $V' := \{\pi(A)\colon A \in V\} \subseteq N$,
- $T' := \{\pi(a)\colon A \in T\} \subseteq N$ and $T' \cap V' = \emptyset$,
- $R' := \{\hat{\pi}(r)\colon r \in R\}$, where

$$\hat{\pi}\colon R \to R',$$
$$\hat{\pi}(A \to X_1 \dots X_k) := \pi(A) \to \pi(X_1) \dots \pi(X_k),$$

- $S' := \pi(S)$,
- P' is determined as

$$P'(r') := \max_{r\colon \hat{\pi}(r) = r'} P(r).$$

The resulting grammar is not always probabilistic because the sum of the values $P'(A \to \zeta)$ for a fixed A can be greater then 1. For each rule $r \in R$ the following relation holds:

$$P'(r') \geq P(r)$$

which guarantees that the probabilities of the rules never decrease during the process of grammar projection. It implies that

$$\alpha_{G',w}(e') \geq \alpha_{G,w}(e)$$

for each edge e and its projection e', so we can use $a(e) = \alpha_{G',w}(e')$ as an estimate.

As in the case of context summary estimates, there is a wide range of possible grammar projection estimates, between the NULL estimate (which corresponds to the constant projection) and the TRUE estimate (based on the identity projection).

III. The implementation of A* parsing algorithm in the Translatica translation system

A. Translatica machine translation system

Translatica is a rule-based machine translation system that translates between languages: Polish, English, German and Russian. The German language parser of the Translatica system (in this paper I will call it *the Translatica parser* for simplicity) uses a weighted context-free grammar and a CYK-based agenda parsing algorithm. The weights used in this parser are probabilities extracted automatically from the TüBa treebank[1] using statistical methods. We decided to try to use the A* parsing algorithm in the Translatica parser to improve its speed and performance.

The implementation of the A* algorithm in the Translatica parser consisted in the adaptation of the current parser to use the A* methods.

B. SX heuristic

One of the heuristics described in [3] is the SX context summary estimate. Let $G = (V, T, R, S, P)$ be a PCFG and $w = w_1 \dots w_n \in T^+$. SX estimate $a_{G,w}(X[i,j])$ specifies the number of terminal symbols to the left from the current edge (i), to the right $(n-j)$ and the label of the considered symbol (X). We chose to use the SX heuristics because [3] predicted that the application of SX heuristics in A* parsing algorithm would result in significant edge savings and require relatively little precomputation.

In every turn of the parser the sum $\beta(e) + a(e)$ is found for each edge e. The value of $\beta(e) + a(e)$ is used to prioritize the edges in agenda. The edge e with the maximal sum $\beta(e) + a(e)$ is removed from the agenda and then processed in the way described before.

The value of the heuristic is calculated only if it is necessary. In the process of memoization, the value of the heuristic for a given context summary is calculated only for the first time and then stored in the memory. This process speeds up the calculations of heuristic.

C. Attribute grammar projection

The grammar used in the Translatica parser is not exactly a PCFG but rather a kind of an attribute grammar. Each rule is equipped with a set of attribute expressions. The rule can be used only if the actual values of the attributes associated with symbols occuring in the rule satisfy the epressions associated with the rule.

It is possible to construct the context-free grammar that is equivalent to a given attribute grammar. This can be obtained by considering all possible configurations of attribute values, fixing the values of attributes and taking the adequate rules into account.

We can project the probabilistic attribute grammar to a PCFG by fixing the values of selected attributes and omitting the others.

We have implemented the grammar projection by fixing the most common attributes.

[1] http://www.sfs.uni-tuebingen.de/en/tuebadz.shtml

TABLE I
AVERAGE TRANSLATION TIMES FOR A SINGLE SET OF 100 SENTENCES.

threshold	50000		10000	
time	absolute	relative	absolute	relative
without A*	1 min 9 s	1.00	36 s	1.00
with A*, SX heur.	2 min 2 s	1.77	52 s	1.44
with A*, NULL heur.	1 min 20 s	1.16	37 s	1.03

TABLE II
AVERAGE TRANSLATION TIMES FOR A SET OF 100 SENTENCES REPEATED 5 TIMES.

threshold	50000		10000	
time	absolute	relative	absolute	relative
without A*	5 min 52 s	1.00	2 m 56 s	1.00
with A*, SX heur.	11 min 50 s	2.02	4 m 32 s	1.55
with A*, NULL heur.	6 min 55 s	1.18	3 m 6 s	1.06

D. NULL heuristic

We have also implemented the NULL heuristics, in order to compare the results of implementing various heuristics. The NULL heuristic, as constatntly equal 0, is easy to implement.

IV. RESULTS

We have conducted various tests in order to compare the performance of the current Translatica parser and the new A*-based parser.

In the Translatica parser it is possible to limit the maximum number of turns of the parser by setting a so called *threshold*. The default value of the threshold in the Translatica parser is 50000.

We have done a series of machine translations (in different conditions) of a sample set of 100 German sentences on various topics. The tests differed in the following parameters:

- the parsing algorithm:
  - the previous Translatica parser,
  - the A* parsing algorithm with SX heuristic,
  - the A* parsing algorithm with NULL heuristic;
- the value of the threshold:
  - 50000,
  - 10000;
- the way the input is given:
  - single sentences,
  - a set of all 100 sentences,
  - a set of all 100 sentences repeated few times.

Differentiation of ways of providing input data is aimed to test the influence of memoization to the performance of the parser.

Table I presents the average times of translating the test set of the 100 German sentences for different threshold values and different heuristics.

Table II presents the average times of translating the test set consisting of 5 copies of the set of 100 German sentences used in the previous test.

The implementation of the A* algorithm applied in the experiment didn't speed up the process of translation in the comparison with the previous Translatica parser. The NULL heuristic was proved to be faster than the SX heuristic and was only slightly slower than the old parser. It is particularly clear for the threshold of 10000, when the translation time using A* with NULL heuristic was comparable to translationa time without A*.

The differences in the quality of the translation were rather insignificant. The majority of the translations were identical, if not, the differences were in general in the single words. There are senteces whose translations by A* algorithm for a threshold of 10000 are identical with translations done by the old algorithm for the threshold of 50000, and better than translations done by the old algorithm for the threshold of 10000. It can mean that the number of steps needed to find the best parse using the A* algorithm is lower then using the previous algorithm. The reason why the translation time with A* is still longer than without A* can be put down to the implementation difficulties. Moreover, it is worth mentioning that the quality of translation of some sentences was better than with the old parser.

V. CONCLUSIONS

The experiment was an innovative try to implement the A* algorithm in an existing CYK-based parser. The implementation was intended to be integrated with the existing parser and to require as little changes to the existing code as possible. The experiment brought a great experience and significant knowledge about the implementational issues of the A* algorithm and about the possibilities of parsing improvement.

There are reasonable grounds that future research will bring the desired results—the faster translation with the Translatica system using the A* algorithm—especially if the following issues will be resolved.

There are various possible reasons why the implementation of A* algorithm in the Translatica machine translation system haven't brought the expected performance improvement.

One of possible reasons can be the specific of the Translatica parser. The grammar used in the Translatica parser doesn't have a distinguished start symbol. A given sentence is parsed using the bottom-up method—the parse tree is built upward while it is possible to combine its edges with edges from the agenda. The advantage of such approach is that it is possible to parse the phrases that are not the full sentences but only the fragments. On the other hand, this solution is rather unfavorable for the implementation possibilities for the A* algorithm because distingushing the start symbol is necessary for proper heuristic calculation.

The other reason is that the fully functional implementation of the A* parsing algorithm in the Translatica parser would involve the significant changes in the structure of the very parser.

VI. FUTURE WORK

We will continue the research concerning the use of modern tools and algorithms to accelerate the process of machine translation. We plan to modify some mechanisms used in the Translatica parser in order to enable the implementation of a

fully functional parser based on the A* algorithm and also some other minor improvements.

REFERENCES

[1] S. A. Caraballo and E. Charniak, "New figures of merit for best-first probabilistic chart parsing," *Computational Linguistics*, vol. 24, 1998, pp. 275-298.

[2] E. Charniak, S. Goldwater and M. Johnson, "Edge-based best-first chart parsing," *In Proceedings of the Sixth Workshop on Very Large Corpora*, 1998, pp. 127-133.

[3] D. Klein and C. D. Manning, "A* Parsing: Fast Exact Viterbi Parse Selection," *In Proceedings of the Human Language Technology Conference and the North American Association for Computational Linguistics (HLT-NAACL)*, 2003, pp. 119-126.

[4] C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*, Prentice Hall, New Jersey, 2000.

[5] A. Ratnaparkhi, "Learning to parse natural language with maximum entropy models," *Machine Learning*, vol. 34, 1999, pp. 151-175.

[6] B. Roark, "Probabilistic top-down parsing and language modeling," *Computational Linguistics*, vol. 27, 2001, pp. 249-276.

[7] P. Skórzewski, *Efektywny parsing języka naturalnego przy użyciu gramatyk probabilistycznych* (*Effective Natural Language Parsing Using Probabilistic Grammars*), Master Thesis on Adam Mickiewicz University, Poznań, 2010.