TAGGING ENGLISH BY PATH VOTING CONSTRAINTS

Gökhan Tür Kemal Oflazer Nihat Özkan
Department of Computer Engineering and Information Science
Bilkent University, Bilkent, Ankara, TR-06533, TURKEY
{tur,ko,nozkan}@cs.bilkent.edu.tr

28 February 1997

Summary

We present a constraint-based tagging approach where individual constraints vote on sequences of matching tokens and tags. Disambiguation of all the tokens in a sentence is performed at the very end by selecting tags that appear on the path that receives highest vote. This constraint application paradigm makes the outcome of the disambiguation *independent* of the rule sequence, and hence relieves the rule developer from worrying about potentially conflicting rule sequencing found in other systems. We have applied this approach to tagging English text from the Energy Abstracts section of the ACL/DCI CD. Our preliminary results indicate that using about 400 constraint rules, we can attain a *recall of 98.2%* and a *precision of 97.1% with about 1.01 parses per token* on previously seen text, and *recall of 95.9%* and a *precision of 93.5% with about 1.025 parses per token* on previously unseen text. Our current system is a prototype implementation, and we propose an efficient implementation based on finite state transducers.

Subject Areas: Constraint-based Tagging, Morphological Disambiguation

Submission Type:

Under consideration for other conference? If Yes, what conference?: No

1. INTRODUCTION

Tagging is one of the preliminary steps in many natural language processing systems in which the proper part-of-speech tag of the tokens comprising the sentences are disambiguated using local information. There has been a large number of studies in tagging using various techniques. Part-of-speech tagging systems have used either a statistical approach where a large corpora has been used to train a probabilistic model which then is used to tag new text, assigning the most likely tag for a given word in a given context (e.g., Church (1988), Cutting et al. (1992), DeRose (1988)). Rule-based or constraint-based approaches recently most prominently exemplified by the Constraint Grammar work (Karlsson et al., 1995; Voutilainen, 1995b; Voutilainen, Heikkilä, and Anttila, 1992; Voutilainen and Tapanainen, 1993), employ a large number of hand-crafted linguistic constraints that are used to eliminate impossible tags or morphological parses for a given word in a given context. Brill (1992; 1994; 1995) has presented a transformation-based learning approach, which induces tagging rules from tagged corpora.

This paper presents a novel approach to constraint-based tagging which relieves the rule developer from worrying about conflicting rule ordering requirements and constraints. The approach depends on assigning votes to constraints and then letting constraints vote on matching sequences on tokens. This approach does not reflect the outcome of matching constraints to the set of morphological parses immediately. Only after all applicable rules are applied to a sentence, all tokens are disambiguated in parallel. Thus, the outcome of the rule applications is independent of the order of rule applications. Rule ordering issue has been discussed by Voutilainen(1994), but he has recently indicated[1] that insensitivity to rule ordering is not a property of their system (although Voutilainen(1995a) states that it is a very desirable property) but rather is achieved by extensively testing and tuning the rules.

[1]Voutilainen, Private communication.

Since tagging has been studied quite extensively we will assume the reader is familiar with the basic problem. Tokens (including punctuation) are assigned via a lexicon or a morphological analyzer, all possible tags from a given tag set. We aim to select one correct tag for each token in a sentence. We consider a token *fully disambiguated* if it has only one tag remaining after tagging. We consider a token as correctly disambiguated, if one of the parses remaining for that token is the *correct* intended parse. We evaluate the resulting disambiguated text by a number of metrics defined as follows (Voutilainen, 1995a):

$$Ambiguity = \frac{\#Parses}{\#Tokens}$$

$$Recall = \frac{\#Tokens\ Correctly\ Disambiguated}{\#Tokens}$$

$$Precision = \frac{\#Tokens\ Correctly\ Disambiguated}{\#Parses}$$

In the ideal case where each token is uniquely and correctly disambiguated with the correct parse, both recall and precision will be 1.0. On the other hand, a text where each token is annotated with all possible parses,[2] the recall will be 1.0, but the precision will be low. The goal is to have both recall and precision as high as possible.

2. TAGGING BY PATH VOTING CONSTRAINTS

This section outlines our approach to constraint-based tagging where constraints vote on matching parses of sequential tokens. We assume that sentences are delineated and that each token is

[2] Assuming no unknown words.

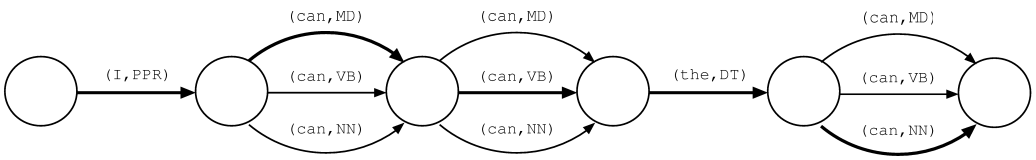

Figure 1: Representing Sentences with a directed acyclic graph

assigned all possible tags by a lexicon or a by a morphological analyzer. We represent each sentence as a standard chart using a directed acyclic graph (DAG) where nodes represent token boundaries and arcs are labeled with ambiguous interpretations of tokens. For instance the sentence

```
I can can the can.
```

would be represented by the graph shown in Figure 1 where bold arcs denote the correct tags.

We describe constraints on the ambiguous interpretation of tokens using rules with two components

$$R = (C_1, C_2, \cdots, C_n; V)$$

where the C_i are, in general, feature constraints on a sequence of the ambiguous parses, and V is an integer denoting the vote of the rule. For English, the features that we use are:

1. `lex`: the lexical form,

2. `tag`: the tag.

It is certainly possibly to extend the set of features used, by including features such as initial letter capitalization, any derivational information, etc.

To illustrate the rules we can give the following examples: examples:

1. `[[TAG=MD][TAG=VB]]` and `[[TAG=MD][TAG=RB][TAG=VB]]` are two constraints which help to disambiguate modal followed a verb possibly interrupted by an adverb.

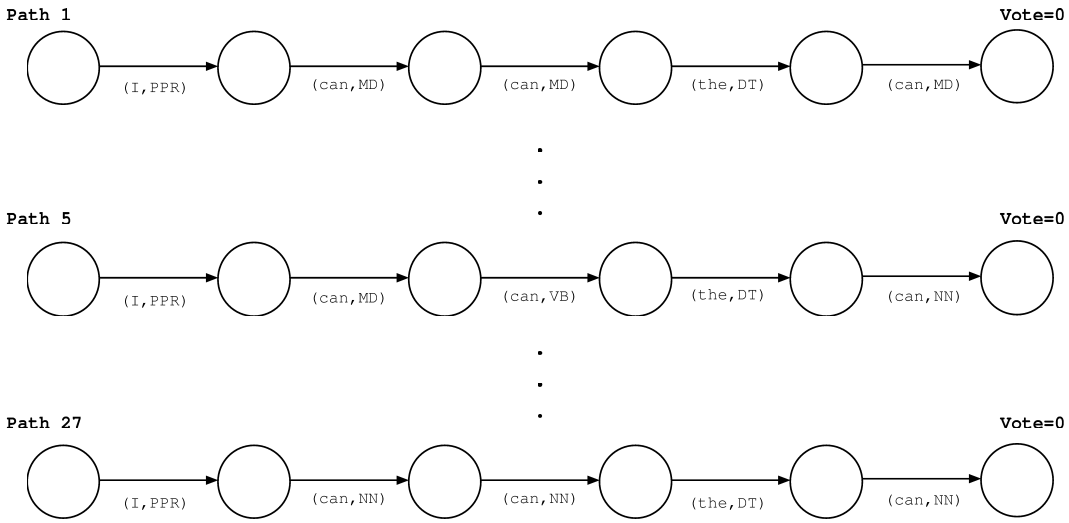

Figure 2: All paths in the Sentence DAG

2. `[[TAG=DT][TAG=JJ][TAG=NN]]` and `[[TAG=DT][TAG=JJ][TAG=NNS]]` are constraints which vote on the simplest noun phrase constructions involving an adjective.

3. `[[TAG=VBG][TAG=VBD]]` is associated with a negative vote hence sequences containing this highly unlikely sequence are discouraged.

4. `[[TAG=DT,LEX=each][TAG=JJ,LEX=other]]` is a rules with a high vote that captures a collocation (Santorini, 1995).

The constraints apply to a sentence in the following manner: Assume for a moment all possible paths from the start node to the end node of a sentence DAG are explicitly enumerated, and that after the enumeration of these paths, paths are augmented by a vote tally component initialized to 0 as depicted in Figure 2.

For each path, we apply each constraint to all possible sequences of token parses. For instance let $R = (C_1, C_2, \cdots, C_n; V)$ be a constraint and let $w_i, w_{i+1}, \cdots, w_{i+n-1}$ be a sequence of token parses labeling sequential arcs of the path. We say R matches this sequence of parses if $w_j, i \leq j \leq i+n-1$ is subsumed by the corresponding constraint $\mathtt{C}_{j-i-1}$. When such a rule matches, the vote of the

path is incremented by V. When all constraints are applied to all possible sequences in all paths, we select the path(s) with the maximum vote. If there are multiple paths with the same maximum vote, the tokens whose parses are different in these paths are assumed to be left ambiguous.

Given that each token has on the average more than 2 possible tags, the procedural description above is very inefficient for all but very short sentences. However, the observation that our constraints are localized to a window of a small number of tokens (say at most 5 tokens in a sequence), suggests a more efficient scheme originally used by Church (1988).[3]

Assume our constraint windows are allowed to look at a window of at most size k sequential parses. Let us take the first k tokens of a sentence and generate all possible paths of k arcs (spanning $k+1$ nodes), and apply all constraints to these "short" paths. Now, if we discard the first token and consider the $(k+1)^{st}$ token, we only need to consider and extend only those paths that have accumulated the maximum vote among paths whose last $k-1$ parses are the same. The reason for this is that since the first token is now out of the context window, it can not influence the application of any rules, hence only the highest scoring (partial) paths need to be extended, as lower scoring paths can not later accumulate votes to surpass the current highest scoring paths.

We can describe the procedure in a slightly more formal way as follows: Let $w_1, w_2, \cdots, w_s$ a the sequence of sentence tokens, $amb(w_i)$ be the number of ambiguous tags for token w_i, and k be the maximum context window size. The procedure then is:

1. $P = \{$ all $\prod_{j=1}^{k-1} amb(w_j)$ paths of the first $k-1$ tokens $\}$
2. $i = k$
3. `while` $i \leq s$
4. `begin`

[3]We will in a later section propose a much more efficient scheme based on finite state transducers.

4.1) Create $amb(w_i)$ copies of each path in P and extend each such copy with one of the distinct tags for token w_i.

4.2) Apply all constraints to the last k tokens of every path in P, updating path votes accordingly.

4.3) Remove from P any path p if there is some other path p' such that $vote(p') > vote(p)$ and the last $k-1$ tags of path p are same as the last $k-1$ tags of p'.

4.4) $i = i + 1$

`end`

2.1. Determining the vote of a rule

There are a number of sources of information for determining the vote of a rule. Intuitively we would like to give high votes to rules that are more specific: i.e., to rules that have higher number of constraints and higher number of features in the constraints. Furthermore we would like to give higher votes to rules which occur as patterns more frequently in a tagged training corpus. In our preliminary results presented in this paper we have manually assigned votes as explained very briefly below but we believe this is not maintainable for a large system and one should resort to machine learning techniques to induce these votes

3. RESULTS FROM TAGGING ENGLISH

We have performed a preliminary evaluation of our approach using the Energy Abstract texts available from the ACL/DCI CD.[4] We developed an initial series of constraint rules using various sources, including Penn Treebank Tagging Guidelines document (Santorini, 1995). A portion of the

[4]We have noted numerous errors and inconsistencies in this corpus and have corrected these wherever possible. Most of the inconsistencies were in direct conflict with the Penn Treebank Tagging Guidelines document (Santorini, 1995).

Tokens	Initial Ambiguity	Final Ambiguity	Recall	Precision
2154	2.051	1.012	98.2%	97.1%
511	2.295	1.025	95.9%	93.5%

Table 1: Tagging Results

Abstracts text was then used as a testbed to find potential holes in our coverage and add further constraints.

For this prototype tagging system we assigned votes to constraints manually giving high votes to constraints that we thought were very strict and less to those that could be violated in certain cases. We also constructed constraints with negative weights to discourage unlikely or invalid sequences. We have noted that the guidelines document suggests rules that depend on certain semantic properties of the words in question or transitivity of the verbs which were not necessarily distinguished by the tag set. We compiled (within a period of about 4 days) about 400 constraint rules with at most 5 constraints in each rule. We will report here two very preliminary results.[5] The first row in Table 1 presents the resulting ambiguity, recall and precision on the text that we used as a testbed or a training corpus. The second line display the results from a short unseen[6] portion of the same corpus.

An analysis of the errors the system makes shows that, most errors stem from the inability to distinguish between the JJ and NN tags of tokens like *local, potential, safe, passive*, etc., and from the inability to distinguish between VBD and VBN readings in cases where the distinguishing piece of information falls out of the 5 token window we have employed. The latter case also accounts for a number of ambiguities remaining. Tokens that appear in bracketing punctuation such as parentheses also account for ambiguities.

[5] We expect to report results on the Brown Corpus in a later version of this paper.
[6] Except for cleaning up of the inconsistencies.

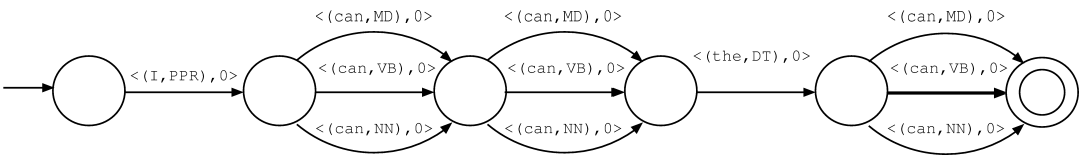

Figure 3: Sentence as a finite state recognizer.

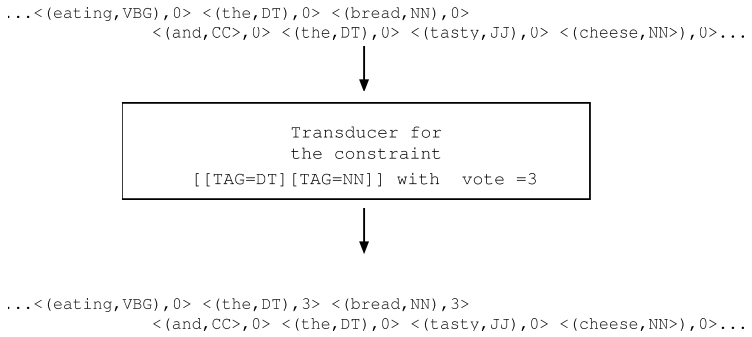

Figure 4: Constraint transducer operating on a tag sequence

4. AN EFFICIENT IMPLEMENTATION TECHNIQUE

The current implementation of the voting approach is meant to be a proof of concept implementation and is a bit inefficient. However, the use of regular relations and finite state transducers (Kaplan and Kay, 1994) provide a very efficient implementation method. For this, we view the DAG representation of a sentence mentioned earlier as a finite state recognizer with the states marking word boundaries and the ambiguous interpretations of the tokens as the state transitions between states, the rightmost node denoting the final state as depicted in Figure 3. The constraints are represented as transducers which increment the votes of the matching transition labels by an appropriate amount[7] (vote of the path divided by the number of constraints in the path so that the effect on the vote of the path is equal to the vote of the path). Such transducers ignore and pass through parses that they are not sensitive to. (See Figure 4 for a depiction.) When a finite state recognizer corresponding to the input sentence (which actually may be considered as an identity

[7]Suggested by Lauri Karttunen (private communication).

transducer) is composed with a constraint transducer, one gets a slightly modified version of the first transducer where the votes of some of the labels have been appropriately incremented and there are possibly additional transitions with incremented labels. When the resulting transducer is composed with all the constraint transducers, all possible votes are cast and the final sentence transducer reflects all the votes. The path in this transducer from the start state to the final state(s) with the highest total label vote can then be selected. The key point here is that due to the nature of the composition operator, the constraint transducers can be composed off-line first, giving a single constraint transducer and then this one is composed with every sentence transducer once (See Figure 5).

5. CONCLUSIONS

We have presented an approach to constraint-based tagging approach which uses constraints, voting on sequences of tokens and tags as its mechanism for tag selection. This approach relieves the rule developer from worrying about rule ordering. Using positive or negative votes we can encourage selection of meaningful sequences of tags or discourage selection of impossible tags. We can also handle lexical biases by using constraint rule of a single constraint that selects the preferred tag for a given token, if no other constraint votes otherwise. Our approach is quite general and is applicable to any language. Our initial results are quite promising and we expect to report results on the Brown Corpus in later version of this paper. We however feel that the current way of assigning weights is not very justifiable and maintainable. We are now experimenting with deriving N-gram frequencies for linguistically meaningful tag sequences from a tagged corpus and modulating the votes of rules derived from their static properties such as number of constraints and features used in the rule, with these frequencies. The proposed approach also is amenable to a high-performance implementation by finite state transducers.

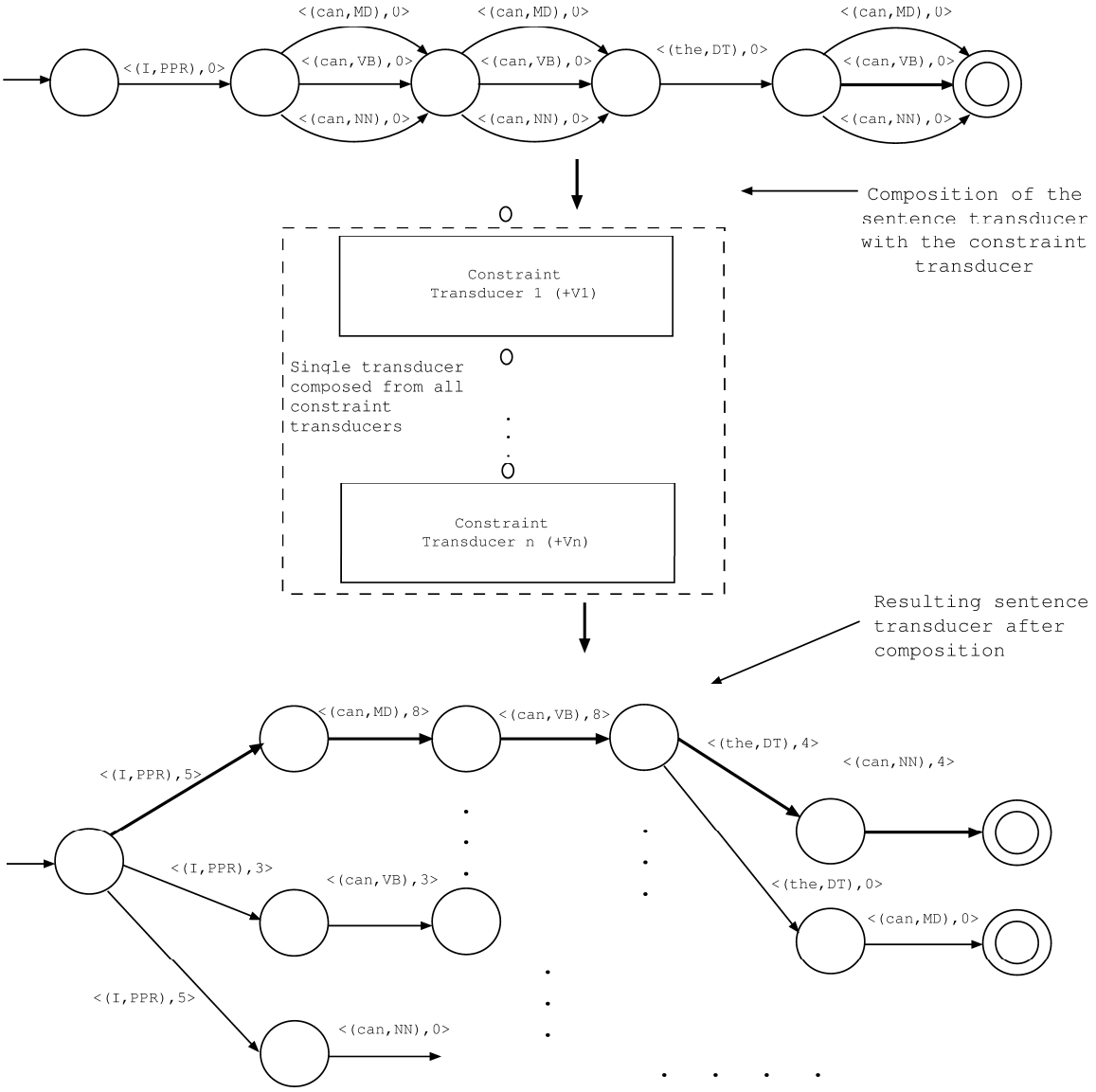

Figure 5: Sentence and Constraint Transducers

6. ACKNOWLEDGMENTS

This research was in part supported by a NATO Science for Stability Grant TU-LANGUAGE.

REFERENCES

Brill, Eric. 1992. A simple-rule based part-of-speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento, Italy.

Brill, Eric. 1994. Some advances in rule-based part of speech tagging. In *Proceedings of the Twelfth National Conference on Articial Intelligence (AAAI-94)*, Seattle, Washinton.

Brill, Eric. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–566, December.

Church, Kenneth W. 1988. A stochastic parts program and a noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, Austin, Texas.

Cutting, Doug, Julian Kupiec, Jan Pedersen, and Penelope Sibun. 1992. A practical part-of-speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento, Italy.

DeRose, Steven J. 1988. Grammatical category disambiguation by statistical optimization. *Computational Linguistics*, 14(1):31–39.

Kaplan, Ronald M. and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–378, September.

Karlsson, Fred, Atro Voutilainen, Juha Heikkilä, and Arto Anttila. 1995. *Constraint Grammar-A Language–Independent System for Parsing Unrestricted Text.* Mouton de Gruyter.

Santorini, Beatrice. 1995. Part-of-speech tagging guidelines fro the penn treebank project. Available at `http://www.ldc.upenn.edu/`. 3rd Revision, 2nd Printing.

Voutilainen, Atro. 1994. *Three studies of grammar-based surface-syntactic parsing of unrestricted English text.* Ph.D. thesis, Research Unit for Computational Linguistics, University of Helsinki.

Voutilainen, Atro. 1995a. Morphological disambiguation. In Fred Karlsson, Atro Voutilainen, Juha Heikkilä, and Arto Anttila, editors, *Constraint Grammar-A Language–Independent System for Parsing Unrestricted Text.* Mouton de Gruyter, chapter 5.

Voutilainen, Atro. 1995b. A syntax-based part-of-speech analyzer. In *Proceedings of the Seventh Conference of the European Chapter of the Association of Computational Linguistics*, Dublin, Ireland.

Voutilainen, Atro, Juha Heikkilä, and Arto Anttila. 1992. *Constraint Grammar of English.* University of Helsinki.

Voutilainen, Atro and Pasi Tapanainen. 1993. Ambiguity resolution in a reductionistic parser. In *Proceedings of EACL'93*, Utrecht, Holland.