A Chart Re-estimation Algorithm for a Probabilistic Recursive Transition Network

Young S. Han*
University of Suwon

Key-Sun Choi†
Korea Advanced Institute of Science and Technology

A Probabilistic Recursive Transition Network is an elevated version of a Recursive Transition Network used to model and process context-free languages in stochastic parameters. We present a re-estimation algorithm for training probabilistic parameters, and show how efficiently it can be implemented using charts. The complexity of the Outside algorithm we present is $O(N^4G^3)$ *where N is the input size and G is the number of states. This complexity can be significantly overcome when the redundant computations are avoided. Experiments on the Penn tree corpus show that re-estimation can be done more efficiently with charts.*

1. Introduction

Though hidden Markov models have been successful in some applications such as corpus tagging, they are limited to the problems of regular languages. There have been attempts to associate probabilities with context-free grammar formalisms. Recently Briscoe and Carroll (1993) have reported work on generalized probabilistic LR parsing, and others have tried different formalisms such as LTAG (Schabes, Roth, and Osborne 1993) and Link grammar (Lafferty, Sleator, and Temperley 1992). Kupiec extended a SCFG that worked on CNF to a general CFG (Kupiec 1991). The re-estimation algorithm presented in this paper may be seen as another version for general CFG.

One significant problem of most probabilistic approaches is the computational burden of estimating the parameters (Lari and Young 1990). In this paper, we consider a **probabilistic recursive transition network** (PRTN) as an underlying grammar representation, and present an algorithm for training the probabilistic parameters, then suggest an improved version that works with reduced redundant computations. The key point is to save intermediate results and avoid the same computation later on. Moreover, the computation of Outside probabilities can be made only on the valid parse space once a chart is prepared.

2. A Probabilistic Recursive Transition Network

A PRTN denoted by λ is a 6-tuple.

$$\lambda \;=\; (\mathcal{A}, \mathcal{B}, \mathcal{S}, \mathcal{F}, \Gamma, \Xi).$$

* Computer Science Department, University of Suwon, Suwon P.O. Box 77–78, Suwon, 440–600, Korea. E-mail: yshan@world.kaist.ac.kr
† Computer Science Department, Korea Advanced Institute of Science and Technology, Taejon, 305-701, Korea. E-mail: kschoi@csking.kaist.ac.kr

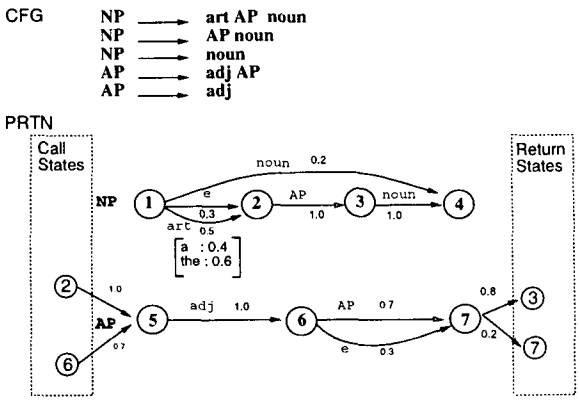

Figure 1
Illustration of PRTN. A parse is composed of dark-headed transitions.

$\mathcal{A}$ is a transition matrix containing transition probabilities, and $\mathcal{B}$ is a word matrix containing the probability distribution of the words observable at each terminal transition. Γ specifies the types of transitions, and Ξ represents a stack. $\mathcal{S}$ and $\mathcal{F}$ denote start and final states, respectively.

Stack operations are associated with transitions; transitions are classified into three types, according to the stack operation. The first type is **nonterminal transition**, in which state identification is pushed into the stack. The second type is **pop transition**, in which transition is determined by the content of the stack. The third type is transitions not committed to stack operation; these are **terminal** and **empty transitions**. In general, the grammar expressed in PRTN consists of layers. A **layer** is a fragment of network that corresponds to a nonterminal. A table of the probability distribution of words is defined at each terminal transition. Pop transitions represent the returning of a layer to one of its (possibly multiple) higher layers.

In this paper, parses are assumed to be sequences of dark-headed transitions (see Figure 1). States at which pop transitions are defined are called **pop states**. Other notations are listed below.

first(l) returns the first state of layer l.

last(l) returns the last state of layer l.

layer(s) returns the layer state s belongs to.

bout(l) returns the states from which layer l branches out.

bin(l) returns the states to which layer l returns.

terminal(l) returns a set of terminal edges in layer l.

nonterminal(l) returns a set of nonterminal edges in layer l.

$\overrightarrow{ij}$ denotes the edge between states i and j.

$[i, j]$ denotes the network segment between states i and j.

$W_{a\sim b}$ is a word sequence covering the a^{th} to b^{th} word.

3. Re-estimation Algorithm

The task of a re-estimation algorithm is to assign probabilities to transitions and the word symbols defined at each terminal transition. The Inside-Outside algorithm provides a formal basis for estimating parameters of context free languages so that the probabilities of the word sequences (sample sentences) may be maximized. The re-estimation algorithm for PRTN uses a variation of the Inside-Outside algorithm customized for PRTN.

Let a word sequence of length N be denoted by:

$$W = W_1 W_2 \cdots W_N .$$

Now define the Inside probability.

Definition 1
The Inside probability denoted by $P_I(i)_{s\sim t}$ of state i is the probability that $layer(i)$ generates the string positioned from s to t starting at state i given a model λ.

That is:

$$P_I(i)_{s\sim t} = P([i,e] \rightarrow W_{s\sim t}|\lambda)$$

where $e = last(layer(i))$. And by definition:

$$P_I(i)_{s\sim t} = \sum_k a_{ik} b(\vec{ik}, W_s) P_I(k)_{s+1\sim t} + \sum_j \sum_{r=s}^{t} a_{ij} a_{uv} P_I(j)_{s\sim r} P_I(v)_{r+1\sim t}. \qquad (1)$$

where $\vec{ik} \in terminal(layer(i))$, $\vec{ij} \in nonterminal(layer(i))$, $u = last(layer(j))$, $v \in bin(layer(j))$, and $layer(i) = layer(v)$.

After the last word is generated, the last state of $layer(i)$ should be reached.

$$P_I(i)_{t+1\sim t} = \begin{cases} 1 & \text{if } i = last(layer(i)), \\ 0 & \text{otherwise.} \end{cases}$$

Figure 2 is the pictorial view of the Inside probability. A valid sequence can begin only at state $\mathcal{S}$, thus to be strict, $P_I(\mathcal{S})$ has an additional product, $P(\mathcal{S})$. When the immediate transition $\vec{ij}$ is of terminal type, the transition probability a_{ij} and the probability of the s^{th} word at the transition $b(\vec{ij}, W_s)$ are multiplied together with the Inside probability of the rest of the sequence, $W_{s+1\sim t}$.

Now define the Outside probability.

Definition 2
The Outside probability denoted by $P_O(i,j)_{s\sim t}$ is the probability that partial sequences, $W_{1\sim s-1}$ and $W_{t+1\sim N}$, are generated, provided that the partial sequence, $W_{s\sim t}$, is generated by $[i,j]$ given a model λ.

And by definition:

$$\begin{aligned} P_O(i,j)_{s\sim t} &= P([\mathcal{S},i] \rightarrow W_{1\sim s-1}, [j,\mathcal{F}] \rightarrow W_{t+1\sim N} \mid \lambda) \\ &= \sum_x \sum_{a=1}^{s} \sum_{b=t}^{N} a_{xf} a_{ey} P_I^{\triangleleft}(f,i)_{a\sim s-1} P_I(j)_{t+1\sim b} P_O(x,y)_{a\sim b}. \qquad (2) \end{aligned}$$

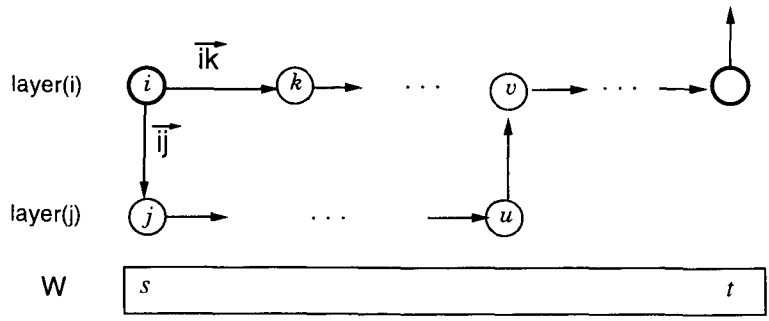

Figure 2
Illustration of Inside probability.

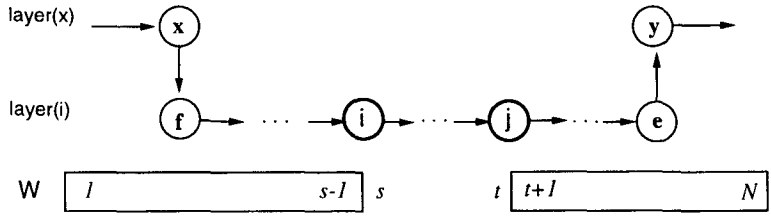

Figure 3
Illustration of Outside probability.

where $x \in bout(layer(i))$, $y \in bin(layer(i))$, $f = first(layer(i))$, $e = last(layer(i))$, $layer(i) = layer(j)$, and $layer(x) = layer(y)$.

The summation on x is defined only when $a \neq 1$ or $b \neq N$ (i.e., there are words left to be generated). Nonterminal and its corresponding pop transitions are defined to be 1 when $a = 1$ and $b = N$.

For a boundary case of the Outside probability where f is the first state of a layer in the above equation:

$$P_O(i,j)_{1\sim N} = \begin{cases} 1 & \text{if } f = \mathcal{S}, \\ 0 & \text{otherwise.} \end{cases}$$

Figure 3 shows the network configuration in computing the Outside probability. In equation 2, $P_I^{\triangleleft}(f,i)_{a\sim s-1}$ is the probability that sequence, $W_{a\sim s-1}$, is generated by $layer(i)$ left to state i, and $P_I(j)_{t+1\sim b}$ is the probability that sequence $W_{t+1\sim b}$ is generated by $layer(i)$ right to state j.

The computation of $P_I^{\triangleleft}(f,i)_{s\sim t}$—a slight variation of the Inside probability in which the $P_I(f)_{a\sim b}$'s in equation 1 are replaced by $P_I^{\triangleleft}(f,i)_{a\sim b}$—is done as follows:

$$P_I^{\triangleleft}(f,i)_{s\sim t} = \begin{cases} P_I(f)_{s\sim t} & \text{if } s \leq t, \\ 1 & \text{if } s > t \text{ and } f = i, \\ 0 & \text{if } s > t \text{ and } f \neq i. \end{cases}$$

It is basically the same as the Inside probability except that it carries an i that indicates a stop state.

Now we can derive the re-estimation algorithm for $\mathcal{A}$ and $\mathcal{B}$ using the Inside and Outside probabilities. As the result of constrained maximization of Baum's auxiliary

function, we have the following form of re-estimation for each transition (Rabiner 1989).

$$\bar{a_{ij}} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}.$$

The expectation of each transition type is computed as follows: For a terminal transition:

$$E_t(\vec{ij}) = \frac{\sum_{r=1}^{N} a_{ij} b(\vec{ij}, W_r) P_O(i,j)_{r\sim r}}{P(W \mid \lambda)}.$$

For a nonterminal transition:

$$E_{nt}(\vec{ij}) = \frac{\sum_{s=1}^{N} \sum_{t=s}^{N} a_{ij} P_I(j)_{s\sim t} a_{uv} P_O(i,v)_{s\sim t}}{P(W \mid \lambda)}.$$

where $u = last(layer(j))$, $v \in bin(layer(j))$, $layer(i) = layer(v)$, $layer(j) = layer(u)$, and $\vec{uv}$ is a pop transition. For a pop transition:

$$E_{pop}(\vec{ij}) = \frac{\sum_{s=1}^{N} \sum_{t=s}^{N} a_{uv} P_I(v)_{s\sim t} a_{ij} P_O(u,j)_{s\sim t}}{P(W \mid \lambda)}.$$

where $u \in bout(layer(i))$, $j \in bin(layer(i))$, $v = first(layer(i))$, $layer(u) = layer(j)$, $layer(v) = layer(i)$, and $\vec{uv}$ is a nonterminal transition.

Since transitions of terminal and nonterminal types can occur together at a state, terminal transitions are estimated as follows:

$$\bar{a_{ij}} = \frac{E_t(\vec{ij})}{\sum_k E_t(\vec{ik}) + \sum_k E_{nt}(\vec{ik})}. \tag{3}$$

For nonterminal transitions:

$$\bar{a_{ij}} = \frac{E_{nt}(\vec{ij})}{\sum_k E_t(\vec{ik}) + \sum_k E_{nt}(\vec{ik})}. \tag{4}$$

And for pop transitions, notice that only pop transitions are possible at a pop state:

$$\bar{a_{ij}} = \frac{E_{pop}(\vec{ij})}{\sum_k E_{pop}(\vec{ik})}. \tag{5}$$

For a terminal transition $\vec{ij}$ and a word symbol w:

$$\bar{b}(\vec{ij}, w) = \frac{\sum_{t \text{ s.t. } W_t = w} a_{ij} b(\vec{ij}, W_t) P_O(i,j)_{t\sim t}}{\sum_{t=1}^{N} a_{ij} b(\vec{ij}, W_t) P_O(i,j)_{t\sim t}}.$$

The re-estimation continues until the probability of the word sequences reaches a certain stability.

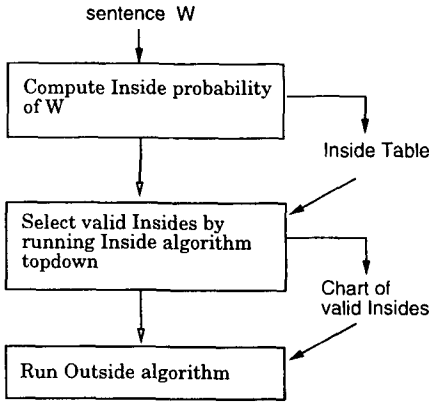

Figure 4
Outside computation with chart. Inside computation builds a table of computed Insides.

4. Chart Re-estimation Algorithm

It can be shown that the complexity of the Inside algorithm is $O(N^3G^3)$ and that of the Outside algorithm is $O(N^4G^3)$ where N is the input size and G is the number of states. The complexity is too much for current workstations when either N or G becomes bigger than a few 10s. A basic implementation of the algorithm is to use a chart and avoid doing the same computations more than once. For instance, the table for storing Inside computations takes $O(N^2G^2C)$ store, where C is the number of terminal and nonterminal categories. A chart item is a function of five parameters, and returns an Inside probability.

$$I(i,j,s,t,c) = P_I(i,j)_{s\sim t}.$$

A chart item is associated with categories implying that the item is valid on the specified categories that begin the net fragment of the item. Suppose a net fragment $[i,j]$ begins with NP and ADJP, then given a sentence fragment $W_{s\sim t}$, ADJP may not participate in generating $W_{s\sim t}$, while NP may. The information of valid categories is useful when the chart is used in computing Outside probabilities.

An Outside probability is the result of computing many Inside probabilities. Computing an Inside probability even in an application of moderate size can be impractical. A naive implementation of Outside computation takes numerous Inside computations, so estimating even a parameter will not be realistic in a serial workstation (Lari and Young 1990).

The proposed estimation algorithm aims at reducing the redundant Inside computations in computing an Outside probability. The idea is to identify the Inside probabilities used in generating an input sentence and to compute an Outside probability using mainly those Insides. This is done first by computing an Inside probability of the input sentence, which can return a table of Insides used in the computation. Note that the Insides in the deepest depth are produced first, as the recursion is released, thus there can be many Insides that are not relevant to the given sentence. The Insides that participate in generating the input sentence can be identified by running the Inside algorithm one more time, top-down. Figure 4 illustrates the steps of the revised Outside computation.

The identified Insides, however, do not cover all the Insides needed in computing an Outside probability. This is because the Inside algorithm works on a network from

left to right and one transition at a time. Many Insides that are missed in the table are compositions of smaller Insides.

Once charts of selected Insides are prepared, an Outside probability is computed as follows:

$$P_O(i,j)_{s\sim t} = \sum_{\{x|I(x,y,a,b,c)>0\}} \sum_{(a,b)\in\sigma(f,e,s,t)} a_{xf}a_{ey}I(f,i,a,s-1)I(j,e,t+1,b)P_O(x,y)_{a\sim b}\,.$$

where $x \in bout(layer(i))$, $y \in bin(layer(i))$, $f = first(layer(i))$, $e = last(layer(i))$, $c \in \{\text{nonterminal}\}$, $layer(i) = layer(j)$, and $layer(x) = layer(y)$.

The function $\sigma(f,e,s,t)$ returns a set of (a,b) pairs where there are Inside items $I(f,e,a,b)$ defined at the chart such that $a \leq s$ and $b \geq t$. In short, the items for state f indicate the possible combinations of sentence segments inclusive of the given fragment $W_{s\sim t}$ because the chart contains items of all the valid sentence segments that were generated through the layer $[f,e]$. When the current layer $[f,e]$ is completed with the two Insides computed, the computation extends to the Outside.

Useless advancements into high layers that do not lead to the successful completion of a given sentence can be avoided by making sure that $[x,y]$ generates $W_{a\sim b}$ and the category of current layer c is defined, which can be checked by consulting the chart items for state x.

5. Experiments

The goal of our experiments is to see how much saving the new estimation algorithm achieves in computational cost. Out of 14,132 Wall Street Journal trees of the Penn tree corpus, 1,543 trees corresponding to sentences with 10 words or less were chosen, and the programs written in C language were run at a Sparc10 workstation.

The basic implementation of an Inside-Outside algorithm assumes tables for Insides and Outsides so that identical Insides and Outsides need not be recomputed. A chart Outside or re-estimation algorithm assumes a refined table of Insides that contains only valid Insides used in generating the input sentence as discussed earlier, and Outside computation is done based on the refined table.

The improvement from the chart re-estimation algorithm is measured in the number of actual Inside and Outside computations done to estimate the parameters. Figure 5 shows the average counts of Insides used in estimating 50 trees randomly selected from 1,543 samples. Before the re-estimation algorithm is applied, an RTN that faithfully encodes the input trees without any overgeneration is constructed from the 50 trees. The gain in Insides from the chart re-estimation algorithm is very clear, and in the case of Outsides the gain is even more conspicuous (see Figure 6). The number of Insides counted in chart version also includes the Insides computed in preparing the chart.

6. Conclusion

We have presented an efficient re-estimation algorithm for a PRTN that made use of only valid Insides. The method requires the preparation of a chart by running Inside computation twice over a whole sentence. The suggested method focuses mainly on reducing the computational overhead of Outside computation, which is a major portion of a re-estimation. The computation of an Inside probability may be improved further using a similar technique introduced in this paper.

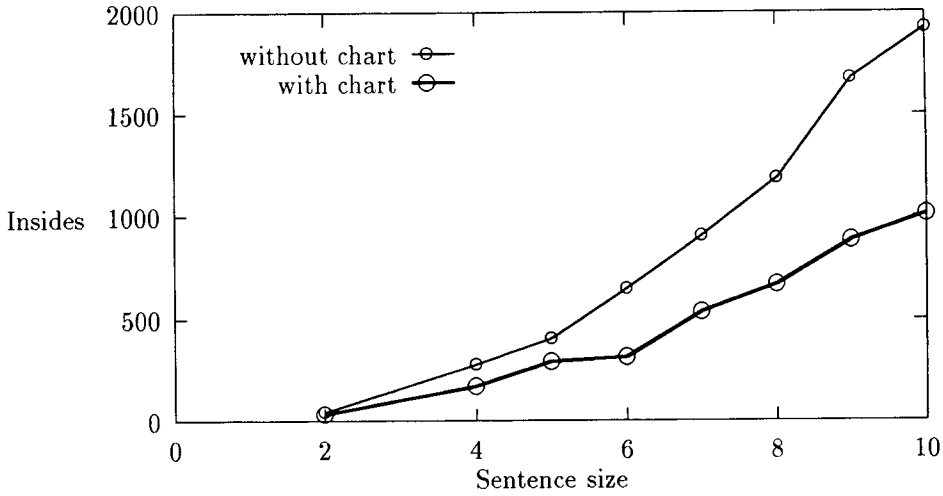

Figure 5
Gain in Insides using chart re-estimation, showing 50 randomly-chosen sentences out of 1,543 samples.

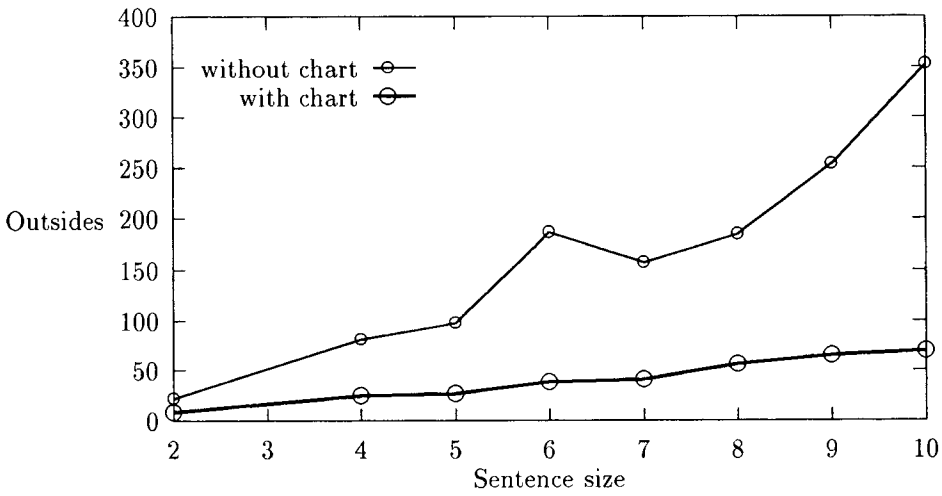

Figure 6
Gain in Outsides using chart re-estimation with the same 50 sentences as in Figure 5.

References

Briscoe, Ted, and John Carroll. 1993. Generalized probabilistic LR parsing of natural language (Corpora) with unification-based grammars." *Computational Linguistics* 19(1): 25–57.

Kupiec, Julian. 1991. A trellis-based algorithm for estimating the parameters of a hidden stochastic context-free grammar. In *Proceedings of the Speech and Natural Language Workshop*, pages 241–246, DARPA, Pacific Grove.

Lafferty, John, Daniel Sleator, and Davy Temperley. 1992. Grammatical trigrams: A probabilistic model of link grammar. *AAAI Fall Symposium Series: Probabilistic Approaches to Natural Language*, pages 89–97, Cambridge.

Lari, K. and S. J. Young. 1990. The estimation of stochastic context-free grammars using the Inside-Outside algorithm. *Computer Speech and Language* 4:35–56.

Rabiner, Lawrence R. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE 77*, Volume 2.

Schabes, Yves, Michael Roth, and Randy Osborne. 1993. Parsing the Wall Street Journal with the inside-outside algorithm. *Sixth Conference of the European Chapter of the ACL, '93*, Utrecht, the Netherlands, April.