Enlarged Search Space for SITG Parsing

Guillem Gascó, Joan-Andreu Sánchez, José-Miguel Benedí
Institut Tecnològic d'Informàtica, Universitat Politècnica de València
Camí de Vera s/n, València, 46022, Spain
ggasco@iti.upv.es , {jandreu,jbenedi}@dsic.upv.es

Abstract

Stochastic Inversion Transduction Grammars constitute a powerful formalism in Machine Translation for which an efficient Dynamic Programming parsing algorithm exists. In this work, we review this parsing algorithm and propose important modifications that enlarge the search space. These modifications allow the parsing algorithm to search for more and better solutions.

1 Introduction

Syntax Machine Translation has received great attention in the last few years, especially for pairs of languages that are sufficiently non-monotonic. Several works have explored the use of syntax for Machine Translation (Wu, 1997; Chiang, 2007). In (Wu, 1997), Stochastic Inverse Transduction Grammars (SITGs) were introduced for describing structurally correlated pairs of languages. SITGs can be used to simultaneously analyze two strings from different languages and to correlate them. An efficient Dynamic Programming parsing algorithm for SITGs was presented in (Wu, 1997). This algorithm is similar to the CKY algorithm for Probabilistic Context Free Grammars. The parsing algorithm does not allow the association of two items that have the empty string in one of their sides. This limitation restricts the search space and prevents the algorithm from exploring some valid parse trees.

In this paper, we review Wu's parsing algorithm for SITGs (referred to as the original algorithm) and propose some modifications to increase the search space in order to make it possible to find these valid parse trees.

2 SITG Parsing

SITGs (Wu, 1997) can be viewed as a restricted subset of Stochastic Syntax-Directed Transduction Grammars (Maryanski and Thomason, 1979). Formally, a SITG in Chomsky Normal Form can be defined as a set of lexical rules that are noted as $A \rightarrow x/\epsilon$, $A \rightarrow \epsilon/y$, $A \rightarrow x/y$; direct syntactic rules that are noted as $A \rightarrow [BC]$; and inverse syntactic rules that are noted as $A \rightarrow \langle BC \rangle$, where A, B, C are non-terminal symbols, x, y are terminal symbols, ϵ is the empty string, and each rule has a probability value p attached. The sum of the probabilities of the rules with the same non-terminal in the left side must be equal to 1. When a direct syntactic rule is used in parsing, both strings are parsed with the syntactic rule $A \rightarrow BC$. When an inverse rule is used in parsing, one string is parsed with the syntactic rule $A \rightarrow BC$, and the other string is parsed with the syntactic rule $A \rightarrow CB$.

An efficient Viterbi-like parsing algorithm that is based on a Dynamic Programming Scheme was proposed in (Wu, 1997). It allows us to obtain the most probable parse tree that simultaneously analyzes two strings, $X = x_1 ... x_{|X|}$ and $Y = y_1 ... x_{|Y|}$, i.e. the bilingual string X/Y. It has a time complexity of $O(|X|^3|Y|^3|R|)$, where $|R|$ is the number of rules of the grammar.

The parsing algorithm is based on the definition of:

$$\delta_{ijkl}(A) = \widehat{\Pr}(A \stackrel{*}{\Rightarrow} x_{i+1} \cdots x_j / y_{k+1} \cdots y_l)$$

as the maximum probability of any parsing tree that simultaneously generates the substrings $x_{i+1} \cdots x_j$ and $y_{k+1} \cdots y_l$ from the non-terminal symbol A .

In (Wu, 1997), the parsing algorithm was defined as follows:

Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL, pages 653–656,
Los Angeles, California, June 2010.

1. Initialization

$$\begin{aligned}
\delta_{i-1,i,k-1,k}(A) &= p(A \to x_i/y_k) \\
&\quad 1 \le i \le |X|, 1 \le k \le |Y|, \\
\delta_{i-1,i,k,k}(A) &= p(A \to x_i/\epsilon) \\
&\quad 1 \le i \le |X|, 0 \le k \le |Y|, \\
\delta_{i,i,k-1,k}(A) &= p(A \to \epsilon/y_k) \\
&\quad 0 \le i \le |X|, 1 \le k \le |Y|,
\end{aligned}$$

2. Recursion

$$\text{For all } A \in N \text{ and } i,j,k,l \text{ such that } \begin{cases} 0 \le i < j \le |X|, \\ 0 \le k < l \le |Y|, \\ j - i + l - k > 2, \end{cases} \quad (1)$$

$$\delta_{ijkl}(A) = \max(\delta^{[]}_{ijkl}(A), \delta^{\langle\rangle}_{ijkl}(A))$$

where

$$\delta^{[]}_{ijkl}(A) = \max_{\substack{B,C \in N \\ i \le I \le j, k \le K \le l \\ (I-i)(j-I)+(K-k)(l-K) > 0}} p(A \to [BC])\delta_{iIkK}(B)\delta_{IjKl}(C) \quad (2)$$

$$\delta^{\langle\rangle}_{ijkl}(A) = \max_{\substack{B,C \in N \\ i \le I \le j, k \le K \le l \\ (I-i)(j-I)+(K-k)(l-K) > 0}} p(A \to \langle BC\rangle)\delta_{iIKl}(B)\delta_{IjkK}(C) \quad (3)$$

This algorithm cannot provide the correct parsing tree in some situations. For example, consider the SITG shown in Fig. 1. If the input pair is a/b,

$$\begin{array}{rlll} p & S & \to [SS] & p \quad S \quad \to \langle SS \rangle \\ q & S & \to \epsilon/b & q \quad S \quad \to a/\epsilon \\ 1-2p-2q & S & \to a/b & \end{array}$$

Figure 1: Example SITG.

this SITG provides the parse tree (a) that is shown in Fig. 2 with probability $1 - 2p - 2q$. However, the parse tree (b) is more likely if $1 - 2p - 2q < 2pq$. The above parsing algorithm is not able to obtain this parse tree due to the restriction $j-i+l-k > 2$ in (1). This restriction does not allow the algorithm to consider two subproblems in which each substring has length 1 which have not been previously considered in the initialization step. Changing this restriction to $j - i + l - k \ge 2$ is not enough to tackle this situation since the restriction

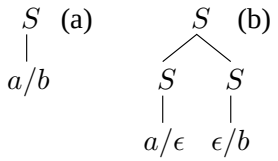

Figure 2: Parse tree (a) can be obtained with Wu's algorithm for a/b, but parse tree (b) cannot be obtained.

$(I-i)(j-I)+(K-k)(l-K) \neq 0$ in expression (2) is not accomplished given that $I = i$ or $I = j$, and $K = k$ or $l = K$ (similarly in expression (3)).

From now on, we will use the term *non-explored trees* to denote the set of trees that are possible when rules of the grammar are applied but cannot be explored with Wu's algorithm. In fact, this situation appears for other paired strings (see Fig. 3) in which a string in one side is associated with the empty string in the other side through rules that are not lexical rules. For example, in Fig. 3b, substring aa could be associated with ϵ. However,this parse tree cannot be obtained with the algorithm due to the search restrictions described above.

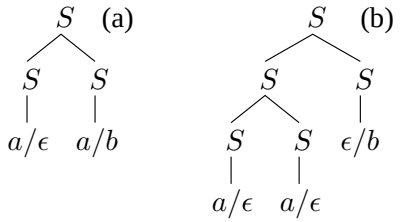

Figure 3: Parse tree (a) can be obtained with Wu's algorithm for aa/b, but parse tree (b) would be more probable if $pq^2 > 1 - 2p - 2q$.

The changes needed in the algorithm to be able to find the sort of parsing trees described above are the following:

- Changing restriction $j-i+l-k > 2$ in (1) to $j-i+l-k \ge 2$. Note that this new restriction is redundant and could be removed.
- Changing restriction $(I-i)(j-I)+(K-k)(l-K) \neq 0$ to $((j-I)+(l-K))*((I-i)+(K-k)) \neq 0$ in (2) and to $((j-I)+(K-k))*((I-i)+(l-K)) \neq 0$ in (3) in order to guarantee the algorithm's termination.

3 Search under SITG Constraints

The modifications that have been introduced in Section 2 enlarge the search space and allow the parsing

algorithm to explore a greater number of possible solutions. We illustrate this situation with an example. Consider the SITG introduced in Figure 1. Fig. 4 shows the possible complete matched trees for the input pair a/b that are considered in the search process with the modifications introduced.

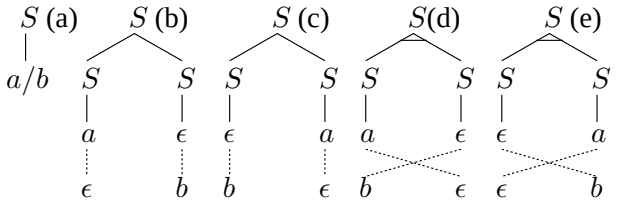

Figure 4: Parse trees for input pair a/b that are taken into account in the search process with the modifications.

Without these modifications, the parsing algorithm only takes into account tree (a) of Fig. 4. For this grammar, we have computed the growth in number of complete matched trees. Table 1 shows how the search space grows notably with the modifications introduced.

n	Wu's alg.	Modified alg.	ratio
1	1	5	0.200
2	34	290	0.117
3	1,928	34,088	0.057
4	131,880	5,152,040	0.026
5	10,071,264	890,510,432	0.011
6	827,969,856	167,399,588,160	0.005

Table 1: Growth in number of explored trees for the original and modified parsing algorithms (n is the length of the input pair strings and the last column represents the ratio between columns two and three).

As a preliminary experiment and in order to evaluate empirically the Wu's parsing algorithm versus the modified algorithm, we parsed first 100K sentence of German-English Europarl corpus. The lexical rules in the Bracketing SITG used for parsing were obtained from a probabilistic dictionary by aligning with IBM3 model (NULL aligments were also included). In this experiment, the modified algorithm obtained a more probable parse tree for 6% of the sentences. If we added brackets to the sentences separately with monolingual parsers, we could use a parsing algorithm similar to the algorithm that is described in (Sánchez and Benedí, 2006). The monolingual brackets restricted the parse tree to those that were compatible with the brackets. In that case the modified algorithm obtained a more probable parse tree for 14% of the sentences.

4 Inside Probability

The parsing algorithm described above computes the most likely parse tree for a given paired string X/Y. However, in some cases (Wu, 1995; Huang and Zhou, 2009), we need the inside probability ($\beta_{0,|X|,0,|Y|}(S)$), i.e., the probability that the grammar assigns to the whole set of parse trees that yield X/Y. If the maximizations are replaced by sums, the algorithm can be used to compute the inside probability. However, as stated above, the original algorithm cannot find the whole set of trees for a given paired string in some cases. These non-explored trees have a probability greater than 0.

As an example, we computed the amount of probability lost in the inside computation using the original algorithm with the grammar shown in Fig. 1. Let Γ be the amount of probability of the non-explored trees (the lost probability). It must be noted that since height 1 trees are all reachable, we must accumulate lost probability for trees of height 2 or more. Hence, let γ be the amount of lost probability for trees of height 2 or more. Note that all such trees must have initially used the production $S \rightarrow SS$ inversely or directly. Thus, $\Gamma = 2p \cdot \gamma$. Fig. 5 shows the kinds of non-explored trees. Then γ is:

$$\gamma = 4{\cdot}q^2{+}2{\cdot}2p{\cdot}(1{-}2p){\cdot}\gamma{+}(2p)^2{\cdot}(2\gamma(1{-}\gamma){+}\gamma^2)$$

The first addend is the probability of the non-explored trees of height 2 (Fig. 5a). The second addend is the probability that one of the subtrees uses a syntactic production, this new subtree produces a non-explored tree ($2p \cdot \gamma$) and the other subtree

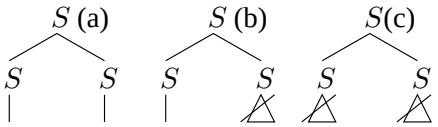

Figure 5: Partial representation of non-explored parse trees from the non-terminal string SS introduced after the first derivation step: (a) both non-terminals yield a terminal in one side and the empty string in the other; (b) one of the non-terminals uses a lexical production and the other non-terminal yields a non-explored tree; (c) both non-terminals use a syntactic production and one (or both) yields a non-explored tree.

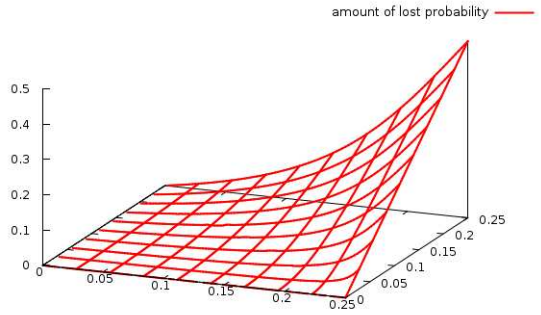

Figure 6: Amount of lost probability for values of p and q.

rewrites itself using a lexical production $(1-2p)$. Note that the non-explored tree can be yielded from either the left or the right non-terminal, (Fig. 5b). The third addend is the probability that both non-terminals use a syntactic production $(2p)^2$ and either one $(2(\gamma)(1-\gamma))$ or both (γ^2) subtrees are non-explored trees (Fig. 5c). If we isolate Γ, we get

$$\Gamma = 2p \cdot \frac{1-4p \pm \sqrt{16p^2-8p+1+64p^2q^2}}{4p^2}$$

Since the solution with the positive square root takes values greater than 1, we can discard it.

Fig. 6 shows the probability accumulated in the non-explored trees for values of p and q between 0 and 0.25 (higher values of p produce inconsistent SITGs). That is the amount of probability lost in the inside parsing for the whole language generated by the grammar shown in Fig. 1.

In order to prove the loss of probability produced by the original algorithm, we use the grammar in Fig. 1 with $p = q = 0.2$. We parse all the paired strings X/Y such that $|X| + |Y| \leq l$, where l is a fixed maximum length. We repeat the same experiment using the modified algorithm. Fig. 7 shows the accumulated inside probabilities for both original and modified algorithms and the theoretical maximums ($1-\Gamma$ for the original algorithm and 1 for the modified algorithm). Note that the computed results approach the theoretical maximums and the modified algorithm covers the whole search space.

5 Conclusions

SITGs have proven to be a powerful tool in Syntax Machine Translation. However, the algorithms have been proposed do not explore all the possible parse trees. This work proposes modifications of the algorithms to be able to explore the whole search space.

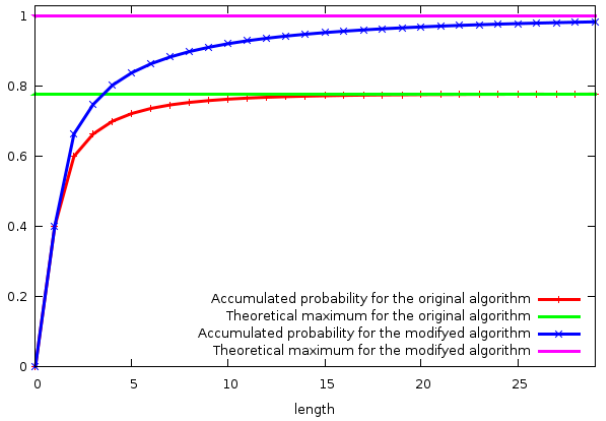

Figure 7: Accumulated inside probability for the original and modified algorithms.

Using an example, we have shown that the modifications allow a complete search. As future work, we plan to proove the correctness of the modified algorithm and to study the impact of these modifications on the use of SITGs for Machine Translation, and the estimation of SITGs.

Acknowledgments

Work supported by the EC (FSE), the Spanish Government (MICINN, "Plan E") under grants MIPRCV "Consolider Ingenio 2010" CSD2007-00018, iTrans2 TIN2009-14511 and the Generalitat Valenciana grant Prometeo/2009/014 and BFPI/2007/117.

References

D. Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

S. Huang and B. Zhou. 2009. An em algorithm for scfg in formal syntax-based translation. In *ICASSP*, pages 4813–4816, Taiwan, China, April.

F.J. Maryanski and M.T. Thomason. 1979. Properties of stochastic syntax-directed tranlation schemata. *Journal of Computer and Information Sciences*, 8(2):89–110.

J.A. Sánchez and J.M. Benedí. 2006. Stochastic inversion transduction grammars for obtaining word phrases for phrase-based statistical machine translation. In *Proc. of Workshop on Statistical Machine Translation. HLT-NAACL 06*, pages 130–133.

D. Wu. 1995. Trainable coarse bilingual grammars for parallel text bracketing. In *Proceedings of the Third Annual Workshop on Very Large Corpora*, pages 69–81.

D. Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404.