Complete Search Space Exploration for SITG Inside Probability

Guillem Gascó, Joan-Andreu Sánchez, and 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 are a very powerful formalism in Machine Translation that allow to parse a string pair with efficient Dynamic Programming algorithms. The usual parsing algorithms that have been previously defined cannot explore the complete search space. In this work, we propose important modifications that consider the whole search space. We formally prove the correctness of the new algorithm. Experimental work shows important improvements in the probabilistic estimation of the models when using the new algorithm.

1 Introduction

Stochastic Inversion Transduction Grammars (SITGs) were introduced in [1] for describing structurally correlated pairs of languages. SITGs can be used to simultaneously analyze two strings from different languages and to correlate them. SITGs have been used in the last few years for Machine Translation (MT), especially for pairs of languages that are sufficiently non-monotonic. Several works have explored its use for MT [2,3,4,5].

An efficient Dynamic Programming parsing algorithm for SITGs was presented in [2]. 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 thus, it prevents exploring some valid parse trees. Expressiveness capacity of SITGs by using Wu's parsing algorithm has been recently studied in [6,7].

In this paper, we propose a new version of the *Inside* parsing algorithm for SITGs that allows to consider all valid parse trees. Then, we also present the formal proof of the correctness, and a set of experiments to demonstrate the usefulness of the new valid parse trees.

2 Inside Probability with SITG

A SITG in Chomsky Normal Form [2] can be defined as a set of lexical rules that are noted as $A \rightarrow a/\epsilon$, $A \rightarrow \epsilon/b$, $A \rightarrow a/b$; direct syntactic rules that are noted as $A \rightarrow [BC]$; and inverse syntactic rules that are noted as $A \rightarrow$

E.R. Hancock et al. (Eds.): SSPR & SPR 2010, LNCS 6218, pp. 306–315, 2010.

$\langle BC\rangle$, where A, B, C are non-terminal symbols, a, b 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 $A \to BC$. When an inverse rule is used in parsing, one string is parsed with the syntactic rule $A \to BC$, and the other string is parsed with the syntactic rule $A \to CB$.

The *inside* probability of a substring pair $(x_{i+1} \dots x_{i+j}, y_{k+1} \dots y_{k+l})$ from the non-terminal symbol A is defined as follows:

$$\mathcal{E}_{i,i+j,k,k+l}[A] = p(A \stackrel{*}{\Rightarrow} x_{i+1} \cdots x_{i+j} / y_{k+1} \cdots y_{k+l}) \ , \tag{1}$$

where j and l represent the size of the subproblems. In this way, the probability of the string pair $(x_1 \dots x_{|x|}, y_1 \dots y_{|y|})$ is $\mathcal{E}_{0,|x|,0,|y|}[S]$.

Let $\mathcal{G}$ be a SITG, and let $(x_1 \dots x_{|x|}, y_1 \dots y_{|y|})$ be a string pair. In general, we can efficiently calculate the probability of this pair by means of a simple modification of the well-known CKY-based *inside* algorithm [8,2]. This algorithm is essentially a Dynamic Programming method, which is based on the construction of a triangular $(n+1)\times(n+1)$ probabilistic parse matrix $\mathcal{E}$. Following a notation very close to [2], each element of $\mathcal{E}$ is a probabilistic nonterminal vector, where their components are computed for all $A \in N$ as:

1. Initialization

$$\mathcal{E}_{i,i+1,k,k+1}[A] = p(A \to x_{i+1}/y_{k+1}) \qquad 0 \le i < |x|\ 0 \le k < |y| \tag{2}$$
$$\mathcal{E}_{i,i+1,k,k}[A] = p(A \to x_{i+1}/\epsilon) \qquad 0 \le i < |x|\ 0 \le k \le |y| \tag{3}$$
$$\mathcal{E}_{i,i,k,k+1}[A] = p(A \to \epsilon/y_{k+1}) \qquad 0 \le i < |x|\ 0 \le k \le |y| \tag{4}$$

2. Recursion

$$\text{For all } A \in N \text{ and } i,j,k,l \text{ such that } \begin{cases} 0 \le i \le |x|,\ 0 \le j \le |x| - i \\ 0 \le k \le |y|,\ 0 \le l \le |y| - k \\ j + l \ge 2, \end{cases} \tag{5}$$

$$\mathcal{E}_{i,i+j,k,k+l}[A] = \mathcal{E}^{[]}_{i,i+j,k,k+l}[A] + \mathcal{E}^{\langle\rangle}_{i,i+j,k,k+l}[A]$$

where

$$\mathcal{E}^{[]}_{i,i+j,k,k+l}[A] = \sum_{\substack{B,C \in N \\ 1 \le I \le j,\ 1 \le K \le l \\ ((j-I)+(l-K))\times(I+K) \ne 0}} p(A \to [BC])\ \mathcal{E}_{i,i+I,k,k+K}[B]\ \mathcal{E}_{i+I,i+j,k+K,k+l}[C] \tag{6}$$

$$\mathcal{E}^{\langle\rangle}_{i,i+j,k,k+l}[A] = \sum_{\substack{B,C \in N \\ 1 \le I \le j,\ 1 \le K \le l \\ ((j-I)K)\times(I+(l-K)) \ne 0}} p(A \to \langle BC\rangle)\ \mathcal{E}_{i,i+I,k+K,k+l}[B]\ \mathcal{E}_{i+I,i+j,k,k+K}[C] \tag{7}$$

The main differences of this algorithm with regard to the Wu's original algorithm are:

- The restriction $j + l \geq 2$ in (5) substitutes the restriction $j + l > 2$ in Wu's algorithm, and
- The restrictions $((j - I) + (l - K)) \times (I + K) \neq 0$ in (6) and $((j - I) + K) \times (I + (l - K)) \neq 0$ in (7) substitute the restriction $I(j - I) + K(l - K) \neq 0$ in Wu's algorithm.

These modifications allow us to consider some parse trees that the original algorithm ignore. Thus, for example, consider a SITG composed by the following rules (the probabilities of the rules have been omited): $(S \rightarrow [SS], S \rightarrow \langle SS \rangle, S \rightarrow \epsilon/b, S \rightarrow a/\epsilon, S \rightarrow a/b)$. If the string pair is (a, b), this SITG could parse this string pair with the parse trees that can be seen in Fig. 1.

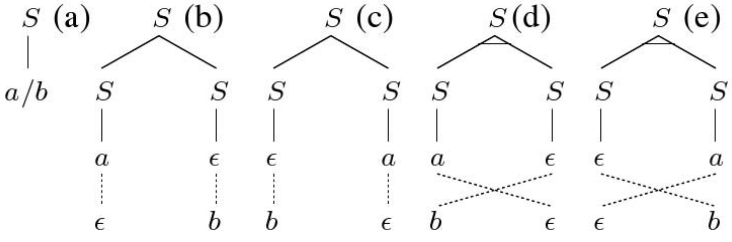

Fig. 1. Parse trees for input pair (a, b) that are taken into account in the search process with the modifications

However, the original algorithm would use just parse tree (a) of Fig. 1. The original algorithm is not able to obtain parse trees (b-e) due to the restriction $j + l > 2$. This restriction does not allow the algorithm to consider subproblems in which each substring has length 1 which have not been previously considered in the initialization step.

In fact, this situation appears for other string pairs (see Fig. 2) 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. 2b, substring *aa* could be associated with ϵ. However, this parse tree cannot be considered with the original algorithm due to the search restrictions that it applied.

Although the modifications to the algorithm allow it to explore more parse trees, the time complexity is the same as in the original algorithm: $O(N^3|x|^3|y|^3)$ where N is the number of non-terminal symbols, $|x|$ is the length of the source language string, and $|y|$ is the length of the target language string.

In order to prove the correctness of the modified algorithm we show that the *inside* probability of a bilingual string computed using it is the correct and complete probability of the string.

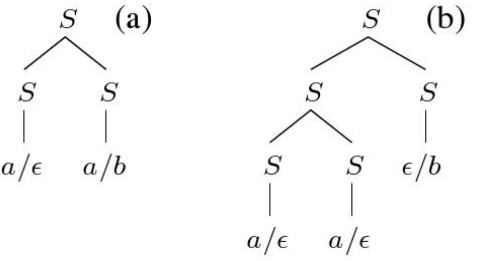

Fig. 2. Parse tree (a) can be obtained with Wu's algorithm for $aa\#b$, but parse tree (b) was not considered

Theorem 1. *If the* inside *algorithm is applied to the string pair* $(x_1 \dots x_{|x|}, y_1 \dots y_{|y|})$ *with a SITG* $\mathcal{G}$*, then the probabilistic parse matrix* $\mathcal{E}$ *collects correctly the probability of this string pair.*

`Proof`

Let $\mathcal{G}$ be a SITG, $p(A \overset{+}{\Rightarrow} x_{i+1} \dots x_{i+j}/y_{k+1} \dots y_{k+l})$ is the inside probability of the substring pair $(x_{i+1} \dots x_{i+j}, y_{k+1} \dots y_{k+l})$.

If the size of subproblems is equal to 1, we can consider the following cases:

- If $j = 1$ and $l = 0$ then, by (3), we have:

$$p(A \overset{+}{\Rightarrow} x_{i+1}/\epsilon) = p(A \to x_{i+1}/\epsilon) = \mathcal{E}_{i,i+1,k,k}[A]$$

  with $0 \le i < |x|$, $0 \le k < |y|$.
- If $j = 0$ and $l = 1$ then, by (4), we have:

$$p(A \overset{+}{\Rightarrow} \epsilon/y_{k+1}) = p(A \to \epsilon/y_{k+1}) = \mathcal{E}_{i,i,k,k+1}[A]$$

  with $0 \le i < |x|$, $0 \le k < |y|$.
- If $j = 1$ and $l = 1$ then, the probability of the substring pair (x_{i+1}, y_{i+1}) is computed with the following possibilities, as illustrated in Fig. 1:

$$\begin{aligned} p(A \overset{+}{\Rightarrow} x_{i+1}/y_{k+1}) = {} & p(A \to x_{i+1}/y_{k+1}) \\ & + \sum_{B,C} p(A \to [BC])p(B \overset{+}{\Rightarrow} x_{i+1}/\epsilon)p(C \overset{+}{\Rightarrow} \epsilon/y_{k+1}) \\ & + \sum_{B,C} p(A \to [BC])p(B \overset{+}{\Rightarrow} \epsilon/y_{k+1})p(C \overset{+}{\Rightarrow} x_{i+1}/\epsilon) \\ & + \sum_{B,C} p(A \to \langle BC \rangle)p(B \overset{+}{\Rightarrow} x_{i+1}/\epsilon)p(C \overset{+}{\Rightarrow} \epsilon/y_{k+1}) \\ & + \sum_{B,C} p(A \to \langle BC \rangle)p(B \overset{+}{\Rightarrow} \epsilon/y_{k+1})p(C \overset{+}{\Rightarrow} x_{i+1}/\epsilon) \end{aligned}$$

with $0 \leq i < |x|$, $0 \leq k < |y|$. Considering the expressions (2), (3) and (4), we have:

$$\begin{aligned} p(A \overset{+}{\Rightarrow} x_{i+1}/y_{k+1}) = {} & \mathcal{E}_{i,i+1,k,k+1}[A] \\ & + \sum_{B,C} p(A \to [BC])\mathcal{E}_{i,i+1,k,k}[B]\mathcal{E}_{i,i,k,k+1}[C] \\ & + \sum_{B,C} p(A \to [BC])\mathcal{E}_{i,i,k,k+1}[B]\mathcal{E}_{i,i+1,k,k}[C] \\ & + \sum_{B,C} p(A \to \langle BC \rangle)\mathcal{E}_{i,i+1,k,k}[B]\mathcal{E}_{i,i,k,k+1}[C] \\ & + \sum_{B,C} p(A \to \langle BC \rangle)\mathcal{E}_{i,i,k,k+1}[B]\mathcal{E}_{i,i+1,k,k}[C] \end{aligned}$$

It is important to note that in Wu's version [2], only the first term is possible, since the rest of the terms are prohibited because it imposes the restriction $j + l > 2$.
Furthermore, in our case, the last 4 terms correspond to the general term of the algorithm for $j = 1, l = 0$ and $j = 0, l = 1$ both for the direct rules and the inverse rules.

– Finally, the possibility $j = 0$ and $l = 0$ is excluded given that there are no rules like $p(A \to \epsilon/\epsilon)$ in the model.

For subproblems of size greater than 1, and in a similar way than in [2], the probability of $p(A \overset{+}{\Rightarrow} x_{i+1} \dots x_{i+j}/y_{k+1} \dots y_{k+l})$ can be solved considering rules (both direct and inverse) and a cutoff points as follows:

$$\begin{aligned} & p(A \overset{+}{\Rightarrow} x_{i+1} \dots x_{i+j}/y_{k+1} \dots y_{k+l}) = \\ & \sum_{B,C} p(A \to [BC]) \\ & \sum_{\substack{1 \leq I \leq j \\ 1 \leq K \leq l}} p(B \overset{+}{\Rightarrow} x_{i+1} \dots x_{i+I}/y_{k+1} \dots y_{k+K}) p(C \overset{+}{\Rightarrow} x_{i+I+1} \dots x_{i+j}/y_{k+K+1} \dots y_{k+l}) \\ & + \sum_{B,C} p(A \to \langle BC \rangle) \\ & \sum_{\substack{1 \leq I \leq j \\ 1 \leq K \leq l}} p(B \overset{+}{\Rightarrow} x_{i+1} \dots x_{i+I+1}/y_{k+K+1} \dots y_{k+l}) p(C \overset{+}{\Rightarrow} x_{i+I+1} \dots x_{i+j}/y_{k+1} \dots y_{k+K}) \end{aligned}$$

With $0 \leq i \leq |x|$, $0 \leq j \leq |x| - i$, $0 \leq k \leq |y|$, $0 \leq l \leq |y| - k$. Considering the definition (1) and the general term of the algorithm, the previous expression can be rewritten as:

$$p(A \overset{+}{\Rightarrow} x_{i+1} \dots x_{i+j} / y_{k+1} \dots y_{k+l}) =$$
$$\sum_{B,C} p(A \to [BC]) \sum_{\substack{1 \le I \le j \\ 1 \le K \le l}} \mathcal{E}_{i,i+I,k,k+K}[B] \mathcal{E}_{i+I,i+j,k+K,k+l}[C]$$
$$+ \sum_{B,C} p(A \to \langle BC \rangle) \sum_{\substack{1 \le I \le j \\ 1 \le K \le l}} \mathcal{E}_{i,i+I,k+K,k+l}[B] \mathcal{E}_{i+I,i+j,k,k+K}[C]$$
$$= \mathcal{E}_{i,i+j,k,k+l}[A]$$

□

Corollary 1. *The probability of the pair string* $(x_1 \dots x_{|x|}, y_1 \dots y_{|y|})$ *can be computed by means of the probabilistic parse matrix* $\mathcal{E}$ *in the following terms:*

$$p(S \overset{+}{\Rightarrow} x_1 \dots x_{|x|} / y_1 \dots y_{|y|}) = \mathcal{E}_{0,|x|,0,|y|}[S]$$

□

3 Experiments

In this section we present several experiments in order to show the performance of the modified SITG parsing algorithm and compare it to the original algorithm. To stress the differences between both algorithms we used the Viterbi parsing algorithm instead of the Inside algorithm. However, similar assumptions can be done for the inside algorithm. Viterbi parsing algorithm computes the most likely parse tree for a given string pair and its probability. The modified version of the Viterbi parsing algorithm [7] can be obtained using maximizations instead of sums in the expressions that have been explained in the previous section.

Experiments were carried out over two different bilingual corpora, the Chinese-English BTEC part of the IWSLT2009 and the French-English Hansard Corpus. The former is a small corpus and we used it to test the differences of both Viterbi parsing algorithms with two languages with a very distinct syntax structure. The later is a larger corpus and is used to test the impact of the modified algorithm for languages with a similar syntactic structure and for large corpora. We explored also the impact of the use of bracketing information in both algorithms. For that purpose we used a parsing strategy similar to the one used in [3], for the corpus partially or fully bracketed. It must be noted that the bracketing information restricts the search to only those parse trees that are consistent with the bracketing. In order to obtain the bracketing information for the corpora, we used several language versions (Chinese, French and English) of the Berkeley Parser [9].

3.1 IWSLT 2009 Corpus

The BTEC part of the IWSLT 2009 corpus [10] is a set of parallel travel sentences in Chinese and English. For the experiments of this work we used the training and the test partitions. Table 1 shows the statistics of this corpus.

As previously mentioned, the original algorithm does not explore all the possible parse trees for a given sentence. In some cases, the algorithm misses the parse tree with the highest probability. As proved in Section 2, the modified algorithm explores the whole search space and, thus, it finds always the most probable tree. In this experiment, we computed the percentage of sentences, for which the parse tree obtained with the modified algorithm have a higher probability than the one obtained with the original algorithm[1]. In other words, the number of times the original algorithm could not explore the best tree. For this purpose, we used a SITG obtained following the method explained in [3]. In addition, for some sentences the original algorithm could not find any parse tree while the modified could.

Table 1. Statistics for IWSLT 2009 Chinese-English BTEC corpus

Corpus Set	Statistic	Chinese	English
Training	Sentences	42,655	
	Words	330,163	380,431
	Vocabulary Size	8,773	8,387
Test	Sentences	511	
	Words	3,352	3,821
	Vocabulary Size	888	813

Table 2 shows the results of the experiment for the non-bracketed corpus (Ch-En), the corpus with only the Chinese side bracketed ([Ch]-En), the corpus with only the English side bracketed (Ch-[En]) and the corpus bracketed in both sides ([Ch]-[En]).

Table 2. Percentage of sentences in IWSLT Corpus for which the original algorithm does not find the parse tree with the highest probability and percentage of sentences not parsed by the original algorithm

Experiment	% of sentences with a different parse tree	% of sentences not parsed with the original algorithm
Ch - En	36.25%	0.24%
[Ch] - En	37.21%	1.4%
Ch - [En]	36.97%	1.02%
[Ch] - [En]	40.93%	3.92%

It must be noted that there was a high percentage of sentences for which the original algorithm could not find the tree with the highest probability. This percentage was even higher when we used bracketing information. The percentage of sentences that could not be parsed with the original algorithm using bracketing in both sides was almost 4%.

[1] Note that the contrary is not possible.

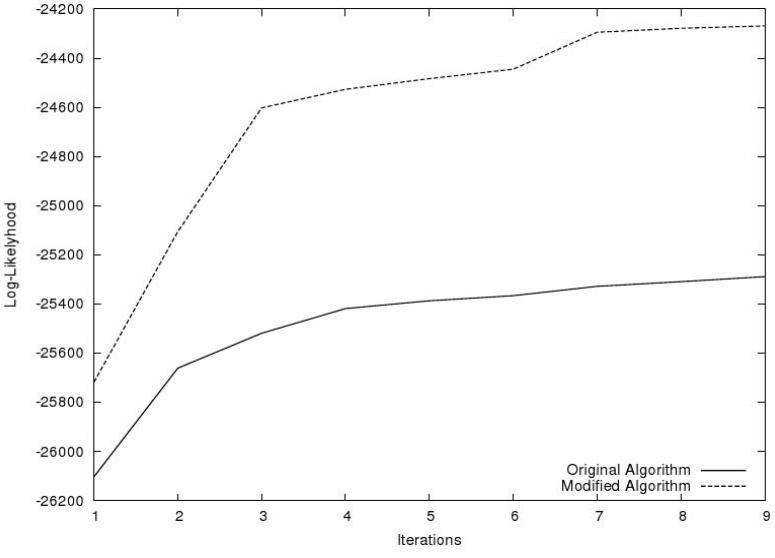

Fig. 3. Log-likelihood of the SITG for several iterations of the Viterbi reestimation using both algorithms on the IWSLT test set

For all the iterations, the reestimation with the modified algorithm results in a more adjusted grammar.

The second experiment tried to determine the importance of the differences between the use of the original or the modified algorithm in the process of SITG reestimation. We performed several iterations of the Viterbi reestimation with each algorithm and we then computed the log-likelihood for the SITG resulting in each iteration over the test set; that is, the logarithm of the product of the probabilities of the SITG parse trees for all the sentences of the test. It is worth noting that for the computation of the log-likelihood we only used those sentences that could be parsed by both algorithms. Figure 3 shows the log-likelihood for each of the iterations and each of the algorithms.

3.2 Hansard Corpus

The Hansard corpus [11] is a set of parallel texts in English and Canadian French, extracted from official records of the Canadian Parliament. Due to the high computational cost of the SITG parsing algorithms and in order to get a faster process, we only used the sentences of length lower than 40 words in each of the languages. The statistics of the resulting corpus are shown in Table 3.

Table 3. Statistics for Hansard French-English corpus (less than 40 words)

Corpus Set	Statistic	French	English
Training	Sentences	997,823	
	Words	16,547,387	14,266,620
	Vocabulary Size	68,431	49,892

Table 4. Percentage of sentences of the Hansard corpus for which the original algorithm does not find the parse tree with the highest probability

Experiment	% of sentences with a different parse tree
Fr - En	27.73%
[Fr] - En	28.06%
Fr - [En]	28.51%
[Fr] - [En]	30.56%

The experiment performed with this corpus is equivalent to the first one explained in the previous subsection. We computed the percentage of times the original algorithm could not find the tree with the highest probability using or not bracketing information. Table 4 shows the results of the experiment.

Compared to the experiment with the IWSLT Corpus, the percentage of sentences with a different parse tree is lower. This behavior may be due to two factors: the similarity in the syntactic structure of both languages and/or the size of the corpus that allows for a better reestimation of the SITG. However, it is still high, almost one third of the sentences of the corpus. The behavior of the algorithms, in respect with the bracketing information is the same as in the IWSLT corpus: the more restricted the search space is, the more differences have the resulting parse trees.

4 Conclusions

SITGs have proven to be a powerful tool in Syntax Machine Translation. However, the parsing algorithms that have been previously proposed do not explore all the possible parse trees. This work propose a modified parsing algorithm that is able to explore the whole search space. We prooved the completeness of the new search. The experiments carried out over two different corpora show that there is a high percentage of sentences for wich the original algorithm cannot find the tree with the highest probability and, in some cases, it cannot find any parse tree at all. In addition, the use of the modified algorithm for reestimation results in better SITGs. As future work, we plan to study the impact of these modifications on the use of SITGs for Machine Translation and the inside-outside SITG reestimation algorithm.

Acknowledgments

Work supported by the EC (FEDER-FSE), the Spanish Government (MICINN, MITyC, "Plan E", under grants MIPRCV "Consolider Ingenio 2010" CSD2007-00018, iTrans2 TIN2009-14511 and erudito.com TSI-020110-2009-439) and the Generalitat Valenciana (grant Prometeo/2009/014 and BFPI/2007/117).

References

1. Wu, D.: Stochastic inversion transduction grammars with application to segmentation, bracketing, and alignment of parallel corpora. In: Proc. of the 14th International Conference on Artificial Intelligence, Montreal, vol. 2, pp. 1328–1335 (1995)
2. Wu, D.: Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. Computational Linguistics 23(3), 377–404 (1997)
3. Sánchez, J., Benedí, J.: Stochastic inversion transduction grammars for obtaining word phrases for phrase-based statistical machine translation. In: Proc. of Workshop on Statistical Machine Translation. HLT-NAACL 2006, New York, USA, June 2006, pp. 130–133 (2006)
4. Huang, S., Zhou, B.: An em algorithm for scfg in formal syntax-based translation. In: ICASSP, Taiwan, China, April 2009, pp. 4813–4816 (2009)
5. Gascó, G., Sánchez, J.A.: Syntax augmented inversion transduction grammars for machine translation. In: Gelbukh, A. (ed.) CICLing 2010. LNCS, vol. 6008, pp. 427–437. Springer, Heidelberg (2010)
6. Soggard, A., Wu, D.: Empirical lower bounds on translation unit error rate for the full class of inversion transduction grammars. In: Proc. 11th Internatnional Conference on Parsing Technologies, Paris, October 2009, pp. 33–36 (2009)
7. Gascó, G., Sánchez, J., Benedí, J.: Enlarged search space for sitg parsing. In: Proc. 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT), Los Angeles, USA (June 2010)
8. Wu, D.: Trainable coarse bilingual grammars for parallel text bracketing. In: Proceedings of the Third Annual Workshop on Very Large Corpora, pp. 69–81 (1995)
9. Petrov, S., Barrett, L., Thibaux, R., Klein, D.: Learning accurate, compact, and interpretable tree annotation. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Sydney, Australia, pp. 433–440. Association for Computational Linguistics (2006)
10. Paul, M.: Overview of the IWSLT 2009 Evaluation Campaign. In: Proc. of the International Workshop on Spoken Language Translation, Tokyo, Japan, pp. 1–18 (2009)
11. Germann, U.: Aligned hansards of the 36th parliament of canada (2001), `http://www.isi.edu/natural-language/download/hansard/`