Some Computational Complexity Results for Synchronous Context-Free Grammars

Giorgio Satta
Dept. of Information Engineering
University of Padua
via Gradenigo, 6/A
I-35131 Padova
Italy
satta@dei.unipd.it

Enoch Peserico
Dept. of Information Engineering
University of Padua
via Gradenigo, 6/A
I-35131 Padova
Italy
enoch@dei.unipd.it

Abstract

This paper investigates some computational problems associated with probabilistic translation models that have recently been adopted in the literature on machine translation. These models can be viewed as pairs of probabilistic context-free grammars working in a 'synchronous' way. Two hardness results for the class NP are reported, along with an exponential time lower-bound for certain classes of algorithms that are currently used in the literature.

1 Introduction

State of the art architectures for machine translation are all based on mathematical models called translation models. Generally speaking, a translation model accounts for all the elementary operations that rule the process of translation between the words and the different word orderings of the source and target languages. Translation models are usually enriched with statistical parameters, to drive the search toward the most likely translation(s). Specialized algorithms are provided for the automatic estimation of these parameters from corpora of translation pairs. Besides the task of natural language translation, statistical translation models are also exploited in other applications, such as word alignment, multilingual document retrieval and automatic dictionary construction.

The most successful translation models that are found in the literature exploit finite-state machinery. The approach started with the so-called IBM models (Brown et al., 1988), implementing a set of elementary operations, such as movement, duplication and translation, that independently act on individual words in the source sentence. These word-to-word models have been later enriched with the introduction of larger units such as phrases; see for instance (Och et al., 1999; Och and Ney, 2002). Still, the generative capacity of these models lies within the realm of finite-state machinery (Kumar and Byrne, 2003), so they are unable to handle nested structures and do not provide the expressivity required to process language pairs with very different word orderings.

Recently, more sophisticated translation models have been proposed, borrowing from the theory of compilers and making use of synchronous rewriting. In synchronous rewriting, two formal grammars are exploited, one describing the source language and the other describing the target language. Furthermore, the productions of the two grammars are paired and, in the rewriting process, such pairs are always applied synchronously. Formalisms based on synchronous rewriting have been empowered with the use of statistical parameters, and specialized estimation and translation (decoding) algorithms were newly developed. Among the several proposals, we mention here the models presented in (Wu, 1997; Wu and Wong, 1998), (Alshawi et al., 2000), (Yamada and Knight, 2001), (Gildea, 2003) and (Melamed, 2003).

In this paper we consider synchronous models based on context-free grammars and probabilistic extensions thereof. This is the most common choice

Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP), pages 803–810, Vancouver, October 2005.

in statistical translation models that exceed the generative power of finite-state machinery. We focus on two associated computational problems that have been defined in the literature. One is the membership problem, which involves testing whether an input string pair can be generated by the model. The other is the translation problem (also called the decoding problem) which involves the search for a suitable translation of an input string/structure. It has been often informally stated in the literature that the use of structured models results in efficient, polynomial time algorithms for the above problems. We show here that sometimes this is not the case. The contribution of this paper can be stated as follows:

- we show that the membership problem is NP-hard, unless a constant bound is imposed on the length of the productions (Section 3);
- we show an exponential time lower bound for the membership problem, in case chart parsing is adopted (Section 3);
- we show that translating an input string into the best parse tree in the target language is NP-hard, even in case productions are bounded in length (Section 4).

Investigation of the computational complexity of translation models has started in (Knight, 1999) for word-to-word models. This paper can be seen as the continuation of that line of research.

2 Synchronous context-free grammars

Several definitions for synchronous context-free grammars have been proposed in the literature; see for instance (Chiang, 2004; Chiang, 2005). Our definition is based on syntax-directed translation schemata (SDTS; Aho and Ullman, 1972), with the difference that we do not impose the restriction that two paired context-free productions have the same left-hand side. As it will be discussed in Section 4, this results in an enriched generative capacity when probabilistic extensions are considered. We assume the reader is familiar with the definition of context-free grammar (CFG) and with the associated notion of derivation.

Let V_N and V_T be sets of nonterminal and terminal symbols, respectively. In what follows we need to represent bijections between all the occurrences of nonterminals in two strings over $V_N \cup V_T$. This can be done by annotating nonterminals with indices from an infinite set. We define $\mathcal{I}(V_N) = \{A^{(t)} \mid A \in V_N,\ t \in \mathbb{N}\}$ and $V_I = \mathcal{I}(V_N) \cup V_T$. We write $\mathsf{index}(\gamma)$, $\gamma \in V_I^*$, to denote the set of all indices (the integers t) that appear in symbols in γ. Two strings $\gamma, \gamma' \in V_I^*$ are **synchronous** if each index in $\mathsf{index}(\gamma)$ occurs only once in γ, each index in $\mathsf{index}(\gamma')$ occurs only once in γ', and $\mathsf{index}(\gamma) = \mathsf{index}(\gamma')$. Therefore synchronous strings have the general form

$$\begin{aligned} &u_{10}A_{11}^{(t_1)}u_{11}A_{12}^{(t_2)}u_{12} \quad \cdots \quad u_{1r-1}A_{1r}^{(t_r)}u_{1r}, \\ &u_{20}A_{21}^{(t_{\pi(1)})}u_{21}A_{22}^{(t_{\pi(2)})}u_{22} \quad \cdots \quad u_{2r-1}A_{2r}^{(t_{\pi(r)})}u_{2r}, \end{aligned}$$

where $r \geq 0$, $u_{1i}, u_{2i} \in V_T^*$, $A_{1i}^{(t_i)}, A_{2i}^{(t_{\pi(i)})} \in \mathcal{I}(V_N)$, $t_i \neq t_j$ for $i \neq j$ and π is some permutation defined on set $\{1, \ldots, r\}$.

Definition 1 *A* **synchronous context-free grammar** *(SCFG) is a tuple $G = (V_N, V_T, P, S)$, where V_N, V_T are finite, disjoint sets of nonterminal and terminal symbols, respectively, $S \in V_N$ is the start symbol and P is a finite set of synchronous productions, each of the form $[A_1 \rightarrow \alpha_1,\ \ A_2 \rightarrow \alpha_2]$, with $A_1, A_2 \in V_N$ and $\alpha_1, \alpha_2 \in V_I^*$ synchronous strings.*

The size of a SCFG G is defined as $|G| = \sum_{[A_1 \rightarrow \alpha_1,\ A_2 \rightarrow \alpha_2] \in P} |A_1 \alpha_1 A_2 \alpha_2|$. Based on an example from (Yamada and Knight, 2001), we provide a sample SCFG fragment translating from English to Japanese, specified by means of the following synchronous productions:

s_1 : [VB $\rightarrow$ PRP$^{(1)}$ VB1$^{(2)}$ VB2$^{(3)}$, VB $\rightarrow$ PRP$^{(1)}$ VB2$^{(3)}$ VB1$^{(1)}$]
s_2 : [VB2 $\rightarrow$ VB$^{(1)}$ TO$^{(2)}$, VB2 $\rightarrow$ TO$^{(2)}$ VB$^{(1)}$ ga]
s_3 : [TO $\rightarrow$ TO$^{(1)}$ NN$^{(2)}$, TO $\rightarrow$ NN$^{(2)}$ TO$^{(1)}$]
s_4 : [PRP $\rightarrow$ he, PRP $\rightarrow$ kare ha]
s_5 : [VB1 $\rightarrow$ adores, VB1 $\rightarrow$ daisuki desu]
s_6 : [VB $\rightarrow$ listening, VB $\rightarrow$ kiku no]
s_7 : [TO $\rightarrow$ to, TO $\rightarrow$ wo]
s_8 : [NN $\rightarrow$ music, NN $\rightarrow$ ongaku]

Note that in production s_2 above, the nonterminals VB and TO generated from nonterminal VB2 in

the English component are inverted in the Japanese component, where some additional lexical material is also added.

In a SCFG, the 'derives' relation is defined on synchronous strings in terms of simultaneous rewriting of two nonterminals with the same index. Some additional notation will help us defining this relation precisely. A **reindexing** is a one-to-one function on $\mathbb{N}$. We extend a reindexing f to V_I by letting $f(A^{(t)}) = A^{(f(t))}$ for $A^{(t)} \in \mathcal{I}(V_N)$ and $f(a) = a$ for $a \in V_T$. We also extend f to strings in V_I^* by letting $f(\varepsilon) = \varepsilon$ and $f(X\gamma) = f(X)f(\gamma)$, for each $X \in V_I$ and $\gamma \in V_I^*$. We say that strings $\gamma_1, \gamma_2 \in V_I^*$ are **independent** if $\mathsf{index}(\gamma_1) \cap \mathsf{index}(\gamma_2) = \emptyset$.

Definition 2 *Let $G = (V_N, V_T, P, S)$ be a SCFG and let γ_1, γ_2 be synchronous strings in V_I^*. The* **derives** *relation $[\gamma_1, \ \gamma_2] \Rightarrow_G [\delta_1, \ \delta_2]$ holds whenever there exist an index t in $\mathsf{index}(\gamma_1)$, a synchronous production $[A_1 \rightarrow \alpha_1, \ A_2 \rightarrow \alpha_2]$ in P and some reindexing f such that*

(i) *$f(\alpha_1\alpha_2)$ and $\gamma_1\gamma_2$ are independent; and*

(ii) *$\gamma_i = \gamma_i' A_i^{(t)} \gamma_i''$, $\delta_i = \gamma_i' f(\alpha_i) \gamma_i''$, for $i = 1, 2$.*

We also write $[\gamma_1, \ \gamma_2] \Rightarrow_G^s [\delta_1, \ \delta_2]$ to explicitly indicate that the derives relation holds through some synchronous production $s \in P$.

Since δ_1 and δ_2 in Definition 2 are synchronous strings, we can define the reflexive and transitive closure of $\Rightarrow_G$, written $\Rightarrow_G^*$. This relation is used to represent derivations in G. In case we have $[\gamma_{1i-1}, \ \gamma_{2i-1}] \Rightarrow_G^{s_i} [\gamma_{1i}, \ \gamma_{2i}]$ for $1 \leq i \leq n$, $n \geq 1$, we also write $[\gamma_{10}, \ \gamma_{20}] \Rightarrow_G^{\sigma} [\gamma_{1n}, \ \gamma_{2n}]$, where $\sigma = s_1 s_2 \cdots s_n$. We always assume some canonical form for derivations (as for instance leftmost derivation on the left component). Similarly to the case of context-free grammars, each derivation in G can be associated with a pair of parse trees, that is, one parse tree for each dimension.

Back to our example, we report a fragment of a derivation of the string pair [he adores listening to music, kare ha ongaku wo kiku no ga daisuki desu]:

$$
\begin{aligned}
&[\mathrm{VB}^{(1)}, \ \mathrm{VB}^{(1)}] \\
&\Rightarrow_G^{s_1} \ [\mathrm{PRP}^{(2)} \ \mathrm{VB1}^{(3)} \ \mathrm{VB2}^{(4)}, \\
&\qquad \mathrm{PRP}^{(2)} \ \mathrm{VB2}^{(4)} \ \mathrm{VB1}^{(3)}] \\
&\Rightarrow_G^{s_4} \ [\mathrm{he} \ \mathrm{VB1}^{(3)} \ \mathrm{VB2}^{(4)}, \\
&\qquad \mathrm{kare \ ha} \ \mathrm{VB2}^{(4)} \ \mathrm{VB1}^{(3)}] \\
&\Rightarrow_G^{s_5} \ [\mathrm{he \ adores} \ \mathrm{VB2}^{(4)}, \\
&\qquad \mathrm{kare \ ha} \ \mathrm{VB2}^{(4)} \ \mathrm{daisuki \ desu}] \\
&\Rightarrow_G^{s_2} \ [\mathrm{he \ adores} \ \mathrm{VB}^{(5)} \ \mathrm{TO}^{(6)}, \\
&\qquad \mathrm{kare \ ha} \ \mathrm{TO}^{(6)} \ \mathrm{VB}^{(5)} \ \mathrm{ga \ daisuki \ desu}].
\end{aligned}
$$

The **translation** generated by a SCFG G is a binary relation over V_T^* defined as

$$
\begin{aligned}
T(G) \ = \ &\{[w_1, \ w_2] \mid [S^{(1)}, \ S^{(1)}] \Rightarrow_G^* [w_1, \ w_2], \\
&w_1, w_2 \in V_T^*\}.
\end{aligned}
$$

The set of strings that are translations of a given string w_1 is defined as:

$$
T(G, w_1) \ = \ \{w_2 \mid [w_1, \ w_2] \in T(G)\}.
$$

A **probabilistic** SCFG (PSCFG) is a pair (G, p_G) where $G = (V_N, V_T, P, S)$ is a SCFG and p_G is a function from P to real numbers in $[0, 1]$ such that, for each $A_1, A_2 \in V_N$, we have:

$$
\sum_{\alpha_1, \alpha_2} p_G([A_1 \rightarrow \alpha_1, \ A_2 \rightarrow \alpha_2] \ = \ 1.
$$

If for $n \geq 1$ and $s_i \in P$, $1 \leq i \leq n$, string $\sigma = s_1 s_2 \cdots s_n$ is a canonical derivation of the form $[S^{(1)}, \ S^{(1)}] \Rightarrow_G^{\sigma} [w_1, \ w_2]$, we write $p_G(\sigma) = \prod_{i=1}^{n} p_G(s_i)$. If $D([w_1, w_2])$ is the set of all canonical derivations in G for pair $[w_1, w_2]$, we write $p_G([w_1, w_2]) = \sum_{\sigma \in D([w_1, w_2])} p_G(\sigma)$.

3 The membership problem

We consider here the **membership** problem for SCFG, defined as follows: for input instance a SCFG G and a pair $[w_1, \ w_2]$, decide whether $[w_1, \ w_2]$ is in $T(G)$. This problem has been considered for instance in (Wu, 1997) for his inversion transduction grammars and has applications in the support of several tasks of automatic annotation of parallel corpora, as for instance segmentation, bracketing, phrasal and word alignment. We show that the membership problem for SCFGs is NP-hard. The result could be derived from the findings in (Melamed et al., 2004) that synchronous rewriting systems as SCFGs are related to the class of so called linear context-free rewriting systems (LCFRSs) and from the result that the membership problem for

LCFRSs is NP-hard (Satta, 1992; Kaji and others, 1994). However, we provide here a direct proof, to simplify the presentation.

Theorem 1 *The membership problem for SCFGs is NP-hard.*

Proof. We reduce from the three-satisfiability problem (3SAT, Garey and Johnson, 1979). Let $\langle U, C\rangle$ be an instance of the 3SAT problem, where $U = \{u_1, \ldots, u_p\}$ is a set of variables and $C = \{c_1, \ldots, c_n\}$ is a set of clauses. Each clause is a set of three literals from $\{u_1, \overline{u}_1, \ldots, u_p, \overline{u}_p\}$.

The general idea of the proof is to use a string pair $[w_1 w_2 \cdots w_p,\ w_c]$, where w_c is a string representation of C and each w_i is a string controlling the truth assignment for the variable u_i. We then construct a SCFG G such that each w_i can be derived in two possible ways only, using some specialized productions of G, encoding the truth assignment of variable u_i. In this way the derivation of the whole string $w_1 \cdots w_p$ in the left dimension corresponds to a guess of a truth assignment for U. Accordingly, on the right dimension only those symbols of w_c will be derived that represent clauses that hold true under the guessed assignment.

We need some additional notation. Below we treat C as an alphabet of atomic symbols. We use a function d such that, for every i with $1 \leq i \leq p$, $c_{d(i,1)}, c_{d(i,2)}, \ldots, c_{d(i,s_i)}$ is the sequence of all clauses that include literal u_i, in the left to right order in which they appear within $c_1 c_2 \cdots c_n$, and $c_{d(i,s_i+1)}, c_{d(i,s_i+2)}, \ldots, c_{d(i,t_i)}$ is the sequence of all clauses that include literal $\overline{u}_i$, again as they appear within $c_1 c_2 \cdots c_n$ from left to right. Note that we must have $\sum_{i=1}^{p} t_i = 3n$. We also use a function e such that, for every $1 \leq i \leq p$ and $1 \leq j \leq t_i$, $e(i,j) = j + \sum_{k=1}^{i-1} t_k$ (assume $\sum_{k=1}^{0} t_k = 0$).

Consider the alphabet $\{a_i, b_i \mid 1 \leq i \leq p\}$. For every i, $1 \leq i \leq p$, let w_i denote a sequence of exactly $t_i + 1$ alternating symbols a_i and b_i, i.e., $w_i \in (a_i b_i)^+ \cup (a_i b_i)^* a_i$. For every $1 \leq i \leq p$, let $x(i,1) = a_i b_i$ and let $x(i,h) = a_i$ (resp. b_i) if h is even (resp. odd), $2 \leq h \leq t_i$. Let also $\overline{x}(i,h) = a_i$ (resp. b_i) if h is odd (resp. even), $1 \leq h \leq t_i - 1$, and let $\overline{x}(i,t_i) = a_i b_i$ (resp. $b_i a_i$) if t_i is odd (resp. even). Therefore we can write $w_i = x(i,1)x(i,2)\cdots x(i,t_1) = \overline{x}(i,1)\overline{x}(i,2)\cdots\overline{x}(i,t_1)$.

Finally, we need a permutation π defined on the set $\{1, \ldots, 3n\}$ as follows. Fix i and j with $1 \leq i \leq p$ and $1 \leq j \leq t_i$, and let h be the number of occurrences of the clause $c_{d(i,j)}$ found in the sequence $c_{d(1,1)}$, $c_{d(1,2)}$, …, $c_{d(1,t_1)}$, $c_{d(2,1)}$, …, $c_{d(i,j)}$. Note that we must have $1 \leq h \leq 3$. Then we set $\pi(e(i,j)) = 3 \cdot [d(i,j) - 1] + h$.

We can now define the target instance $\langle G, [w, w']\rangle$ of our reduction. Let $[w, w'] = [w_1 w_2 \cdots w_p,\ c_1 c_2 \cdots c_n]$. Let also $G = (V_N, V_T, P, S)$, with $V_N = \{S\} \cup \{A_i \mid 1 \leq i \leq 3n\}$ and $V_T = C \cup \{a_i, b_i \mid 1 \leq i \leq p\}$. The productions below define set P:

(i) for every $1 \leq i \leq p$:

(a) for $1 \leq h \leq s_i$:

$$[A_{e(h,i)} \to x(i,h),\ A_{e(h,i)} \to c_{e(i,h)}],$$
$$[A_{e(h,i)} \to x(i,h),\ A_{e(h,i)} \to \varepsilon],$$
$$[A_{e(h,i)} \to \overline{x}(i,h),\ A_{e(h,i)} \to \varepsilon];$$

(b) for $s_i + 1 \leq h \leq t_i$:

$$[A_{e(h,i)} \to x(i,h),\ A_{e(h,i)} \to \varepsilon],$$
$$[A_{e(h,i)} \to \overline{x}(i,h),\ A_{e(h,i)} \to c_{e(i,h)}],$$
$$[A_{e(h,i)} \to \overline{x}(i,h),\ A_{e(h,i)} \to \varepsilon];$$

(ii) $[S \to A_{e(1,1)}^{(e(1,1))} A_{e(1,2)}^{(e(1,2))} \cdots A_{e(1,t_1)}^{(e(1,t_1))} A_{e(2,1)}^{(e(2,1))} \cdots A_{e(p,t_p)}^{(e(p,t_p))}$,

$S \to A_{\pi(e(1,1))}^{(\pi(e(1,1)))} A_{\pi(e(1,2))}^{(\pi(e(1,2)))} \cdots A_{\pi(e(1,t_1))}^{(\pi(e(1,t_1)))} A_{\pi(e(2,1))}^{(\pi(e(2,1)))} \cdots A_{\pi(e(p,t_p))}^{(\pi(e(p,t_p)))}]$.

It is easy to see that $|G|$, $|w|$ and $|w'|$ are polynomially related to $|U|$ and $|C|$. From a derivation of $[w, w'] \in T(G)$, we can exhibit a truth assignment that satisfies C simply by reading off the derivation of the left string $w_1 w_2 \cdots w_p$. Conversely, starting from a truth assignment that satisfies C we can prove $w \in L(G)$ by means of (finite) induction on $|U|$: this part requires a careful inspection of all items in the definition of G. ∎

From Theorem 1 we may conclude that algorithms for the membership problem for SCFGs are very unlikely to run in polynomial time. In the literature, several algorithms for this problem have been proposed using tabular methods (chart parsing). In the worst case, all these algorithms run in time $\Theta(|G| \cdot n^{k(G)})$, with G an SCFG and n the

length of the input string pair. We know that, unless P = NP, $k(G)$ cannot be a constant. We now prove a lower bound on $k(G)$, providing thereby an exponential time lower bound result for our problem under the assumption of the tabular paradigm.

Tabular methods for the membership problem are based on the following representation. Given a synchronous production

$$s: \quad [A_1 \rightarrow B_{11}^{(1)} \cdots B_{1r}^{(r)}, \; A_2 \rightarrow B_{21}^{(\pi(1))} \cdots B_{2r}^{(\pi(r))}], \qquad (1)$$

the already recognized constituent pairs $B_{1i}, B_{2\pi(i)}$ are gather together in several steps, keeping a record of the spanned substrings of the input. To provide a concrete example, if we gather all the B_{1i}'s on the left dimension from left to right, the partial analysis we obtain after the first step can be represented as a **state** $\langle s(1), (i_{11}, j_{11}), (i_{21}, j_{21})\rangle$, meaning that B_{11} and $B_{2\pi(1)}$ span substrings $w_1[i_{11}, j_{11}]$ and $w_2[i_{21}, j_{21}]$, respectively.[1] At the second step we have a state $\langle s(2), (i_{11}, j_{12}), (i_{21}, j_{21}), (i_{22}, j_{22})\rangle$, meaning that $B_{11}B_{12}$ together span $w_1[i_{11}, j_{12}]$, $B_{2\pi(1)}$ spans $w_2[i_{21}, j_{21}]$ and $B_{2\pi(2)}$ spans $w_2[i_{22}, j_{22}]$. We can see that, for some worst case permutations, the left-to-right strategy demands for increasingly more pairs of indices, so that the exponent in the time complexity linearly grows with r.

How much better can we do, if we exploit some strategy other than the left-to-right above? More precisely, we ask how many unconnected spannings a state may require for some worst case permutation π, under the choice of the best possible parsing strategy for π itself.

Theorem 2 *In the worst case, standard tabular methods for the SCFG membership problem require an amount of time $\Omega(|G|\, n^{c\cdot\sqrt{r}})$, with r the length of the longest production in G and c a constant.*

Proof. For any $r \geq 8$ we let $q = \lfloor\sqrt{r/2}\rfloor \geq \lfloor\sqrt{8/2}\rfloor = 2$, and define a permutation π_r on $\{1, \ldots, r\}$. We view the domain of π_r as composed of $2q$ **blocks** with q adjacent integers each, possibly followed by $r - 2q^2$ additional "padding" integers, and its codomain as composed of q blocks with $2q$ adjacent integers each, again possibly followed by $r - 2q^2$ "padding" integers. Permutation π_r transposes all blocks by sending the j-th element of the i-th block in the domain into the i-th element of the j-th block in the codomain, while mapping each padding integer identically into itself. Formally, for all positive integers $i \leq 2q$ and $j \leq q$, $\pi_r(q \cdot (i-1) + j) = 2q \cdot (j-1) + i$, and for all integers i with $2q^2 < i \leq r$, $\pi_r(i) = i$.

We count below how many spans are instantiated by a state that has gathered p constituent pairs, $1 \leq p \leq r$, in parsing production (1) under any possible strategy. When a constituent pair $B_{1i}, B_{2\pi_r(i)}$ is gathered, we say integer i in the domain of π_r and integer $\pi_r(i)$ in the codomain have been pebbled. In this way each span (i, j) in a state corresponds to some run $i, i+1, \ldots j$ of pebbled integers, with either $i = 1$ or $i - 1$ unpebbled, and with either $j = r$ or $j + 1$ unpebbled. We call each such run a **segment**, and show that every parsing strategy demands at least $q = \lfloor\sqrt{r/2}\rfloor$ segments either in the domain or in the codomain of π_r.

We say that a block in the domain of π_r is empty, full, or mixed if, respectively, none, all, or some but not all of its elements have been pebbled. Assume that, for a given parsing strategy, the last block that becomes mixed does so when we place the i-th pebble, and the first block that becomes full does so when we place the j-th pebble. Obviously $i \neq j$: the first pebble placed in a previously empty block can not make it full since every block contains at least 2 elements.

If $i < j$, after placing the i-th pebble and before placing the j-th pebble every block in the domain of π_r is mixed. Each of these $2q$ blocks then contains at least one pebbled element which is adjacent to an unpebbled one and must therefore be either the first or the last element of a segment. The domain of π_r then contains at least $2q/2 = q$ segments.

If $j < i$, after placing the j-th pebble and before placing the i-th pebble at least one block in the domain of π_r (e.g., the h-th block) is full, and at least one (e.g., the k-th) is empty. Then, in each of the q blocks in the codomain of π_r, the h-th element is pebbled while the k-th is not. Therefore the h-th elements of any two consecutive blocks in the codomain of π_r must belong to two distinct segments, since at least one intermediate element is not

[1] For a string $w = a_1 \cdots a_n$, we write $w[i, j]$ to denote the substring $a_{i+1} \cdots a_j$.

pebbled. The codomain of π_r then contains at least q segments. ■

4 The translation problem

In this section we consider some formulations of the translation problem for PSCFG that have been proposed in the literature. The most general definition of the **translation problem** for PSCFG is this: for an input PSCFG $G_p = (G, p_G)$ and an input string w, produce a representation of all possible parse trees, along with their probabilities, that are assigned by G to a string in the set $T(G, w)$ under some translation of w.

Variant of this definition can be found where the input is a single parse tree for w (Yamada and Knight, 2001), or where the output is a single parse tree, chosen according to some specific criteria (Wu and Wong, 1998). To formally study these problems, in what follows we focus on single parse trees associated with derivations in G_p. For a derivation σ of the form $[S^{(1)}, S^{(1)}] \Rightarrow_G^{\sigma} [w_1, w_2]$, we write $t_{\sigma,l}$ and $t_{\sigma,r}$ to denote the left and the right parse trees, respectively, associated with σ. The probability that $t_{\sigma,r}$ is obtained as a translation of $t_{\sigma,l}$ through G_p is thus $p_G([t_{\sigma,l},\ t_{\sigma,r}]) = p_G(\sigma)$. Let t be some parse tree; we write $y(t)$ to denote the string in the yield of t. For a string $w \in V_T^*$ and a parse tree t, we also consider the probability that t is obtained from w through G_p, defined as:

$$p_G([w,\ t]) \quad = \sum_{y(t')=w} p_G([t',\ t]). \tag{2}$$

We can now precisely define the variants of the translation problem we are interested in. Given as input a PSCFG $G_p = (G, p_G)$ and two strings $w_1, w_2 \in V_T^*$, output the pair of parse trees

$$\operatorname*{argmax}_{\substack{y(t_1)=w_1,\\ y(t_2)=w_2}} p_G([t_1,\ t_2]). \tag{3}$$

If the synchronous productions in the underlying SCFG G have length bounded by some constant, then the above problem can be solved in polynomial time using extensions of the Viterbi search strategy to parse forests. This has been shown for instance in (Wu and Wong, 1998; Yamada and Knight, 2001; Melamed, 2004).

A second interesting problem is defined as follows. Given as input a PSCFG $G_p = (G, p_G)$ and a string $w \in V_T^*$, output the parse tree

$$\operatorname*{argmax}_t\ p_G([w,\ t]). \tag{4}$$

Even in case we impose some constant bound on the length of the synchronous productions in G, the above problem is NP-hard, as we show in what follows.

We assume the reader is familiar with the definition of probabilistic context-free grammar (PCFG) and with the associated notion of derivation probability (Wetherell, 1980). We denote a PCFG as a pair (G, p_G), with $G = (V_N, V_T, P, S)$ the underlying context-free grammar and p_G the associated function providing the probability distributions for the productions in P, conditioned on their left-hand side. A probabilistic regular grammar (PRG) is a PCFG with underlying productions of the form $A \to aB$ or $A \to \varepsilon$, with A, B nonterminal symbols and a a terminal symbol.

We consider below a decision problem associated with PRG, called the **consensus** problem, defined as follows: Given as input a PRG (G, p_G) and a rational number $d \in [0, 1]$, decide whether there exists a string w in the language generated by G such that $p_G(w) \geq d$. It has been shown in (Casacuberta and de la Higuera, 2000) that, for a PRG G whose productions have all probabilities expressed by rational numbers, the above problem is NP-complete. (Essentially the same result is also reported in (Lyngso and Pedersen, 2002), stated in terms of hidden Markov models.) We reduce the consensus problem for PRG to a decision version of the problem in (4), called the **best translated derivation** problem and defined as follows. Given as input a PCFG $G_p = (G, p_G)$, a string $w \in V_T^*$ and a rational number $d \in [0, 1]$, decide whether $\max_t\ p_G([w,\ t]) \geq d$.

Theorem 3 *The best translated derivation problem for the class PSCFG is NP-hard.*

Proof. We provide a reduction from the consensus problem for the class PRG with rational production probabilities. The main idea is described in what follows. Given the input PRG G_p, we construct a target PSCFG G'_p that translates string \$ into \$, with \$ a special symbol. Given as input the string \$, G'_p simulates all possible derivations of G_p through its own

derivations. This is done by encoding the nonterminals appearing in a derivation ρ of G_p within the left component of some derivation σ of G'_p, and by encoding the terminal string generated by ρ within the right component of σ. The probability of ρ is also preserved by σ.

Let $G_p = (G, p_G)$, d be an instance of the consensus problem as above, with $G = (V_N, V_T, P, S)$. We specify a PSCFG $G'_p = (G', p_{G'})$ with $G' = (V'_N, \{\$\}, P', S)$ and $V'_N = V_N \cup V_T$. Set P' is constructed as follows:

(i) for every $(S \to aA) \in P$, $s : [S \to A^{(1)},\ S \to a^{(1)}]$ is added to P', with $p_{G'}(s) = p_G(S \to aA)$;

(ii) for every $(S \to \varepsilon) \in P$, $s : [S \to \$,\ S \to \$]$ is added to P', with $p_{G'}(s) = p_G(S \to \varepsilon)$;

(iii) for every $a \in V_T$ and $(A \to bB) \in P$, $s : [A \to B^{(1)},\ a \to b^{(1)}]$ is added to P', with $p_{G'}(s) = p_G(A \to bB)$

(iv) for every $a \in V_T$ and $(A \to \varepsilon) \in P$, $s : [A \to \$,\ a \to \$]$ is added to P', with $p_{G'}(s) = p_G(A \to \varepsilon)$.

Note that the construction of G'_p can be carried out in quadratic time in the size of G_p. It is not difficult to see that there exists a derivation of the form $S \Rightarrow_G a_1 A_1 \Rightarrow_G a_1 a_2 A_2 \cdots \Rightarrow_G a_1 a_2 \cdots a_n A_n$ if and only if there exist a derivation in G' associated with unary trees t_1 and t_2, such that string $SA_1A_2 \cdots A_n$ is read from the spine of t_1 and string $Sa_1a_2 \cdots a_n$ is read from the spine of t_2. Furthermore, the two derivations are composed of 'corresponding' productions with the same probabilities. We conclude that there exists a string w in $L(G)$ with $p_G(w) > d$ if and only if there exists a unary tree t with string $Sw\$$ read from the spine such that $p_{G'}([\$, t]) > d$. ■

We discuss below an interesting consequence of Theorem 3. The SDTS formalism discussed in Section 1 has been extended to the probabilistic case in (Maryanski and Thomason, 1979), called stochastic SDTS (SSDTS). As a corollary to the proof of Theorem 3, we obtain that one can define, through some PSCFG G_p and some fixed string w, a probability distribution $p_G([w, t])$ on parse trees that cannot be obtained through any SSDTS. Without providing the details of the definition of SSDTS, we give here only an outline of the proof. We also assume that the reader is familiar with probabilistic finite automata and with their distributional equivalence with PRG.

Consider the PSCFG $G'_p = (G', p_{G'})$ defined in the proof of Theorem 3, and assume there exists some SSDTS $G''_p = (G'', p_{G''})$ such that, for every tree t, we have $p_{G''}([\$, t]) = p_{G'}([\$, t])$. Since in a derivation of an SDTS the generated trees are always isomorphic, up to some reordering of sibling nodes, we obtain that the productions of G'' must have the form $[S \to a^{(1)},\ S \to a^{(1)}]$, $[a \to b^{(1)},\ a \to b^{(1)}]$ and $[a \to \$,\ a \to \$]$. From these productions we can construct a probabilistic deterministic finite automaton generating the same language as the PRG G_p, and with the same distribution. But this is impossible since there are string distributions defined by some PRG that cannot be obtained through probabilistic deterministic finite automata; see for instance (Vidal et al., 2005).

We conclude by remarking that in (Casacuberta and de la Higuera, 2000) it is shown that finding the best output string for a given input string is NP-hard for stochastic SDTS with a single nonterminal in each production's right-hand side. Our result in Theorem 3, stated for PSCFG, is stronger, since it investigates individual parse trees rather than strings.

5 Concluding remarks

The presented results are based on worst case analysis: further experimental evaluation needs to be carried out on multilingual corpora in order to asses the practical impact of these findings.

Acknowledgment

We are indebted to Dan Melamed and Mark-Jan Nederhof for technical discussion on topics related to this paper. Dan Melamed also suggested to us the problem investigated by Theorem 2. The first author is partially supported by MIUR under project PRIN No. 2003091149_005.

References

A. V. Aho and J. D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

Hiyan Alshawi, Srinivas Bangalore, and Shona Douglas. 2000. Learning dependency translation models as collections of finite state head transducers. *Computational Linguistics*, 26(1):45–60, March.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, Robert L. Mercer, and Paul Roossin. 1988. A statistical approach to language translation. In *Proceedings of the International Conference on Computational Linguistics (COLING) 1988*, pages 71–76, Budapest, Hungary, August.

F. Casacuberta and C. de la Higuera. 2000. Computational complexity of problems on probabilistic grammars and transducers. In L. Oliveira, editor, *Grammatical Inference: Algorithms and Applications; 5th International Colloquium, ICGI 2000*, pages 15–24. Springer.

D. Chiang. 2004. *Evaluating Grammar Formalisms for Applications to Natural Language Processing and Biological Sequence Analysis*. Ph.D. thesis, Department of Computer and Information Science, University of Pennsylvania.

D. Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. of the 43rd ACL*, pages 263–270.

M. R. Garey and D. S. Johnson. 1979. *Computers and Intractability*. Freeman and Co., New York, NY.

Daniel Gildea. 2003. Loosely tree-based alignment for machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Sapporo, Japan, July.

Y. Kaji et al. 1994. The computational complexity of the universal recognition problem for parallel multiple context-free grammars. *Computational Intelligence*, 10(4):440–452.

Kevin Knight. 1999. Decoding complexity in word-replacement translation models. *Computational Linguistics, Squibs and Discussion*, 25(4).

S. Kumar and W. Byrne. 2003. A weighted finite state transducer implementation of the alignment template model for statistical machine translation. In *Proceedings of HLT-NAACL*.

R. B. Lyngso and C. N. S. Pedersen. 2002. The consensus string problem and the complexity of comparing hidden markov models. *Journal of Computing and System Science*, 65:545–569.

Fred J. Maryanski and Michael G. Thomason. 1979. Properties of stochastic syntax-directed translation schemata. *International Journal of Computer and Information Sciences*, 8(2):89–110.

I. Dan Melamed, Giorgio Satta, and Benjamin Wellington. 2004. Generalized multitext grammars. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, Barcelona, Spain.

I. Dan Melamed. 2003. Multitext grammars and synchronous parsers. In *Proceedings of the Human Language Technology Conference and the North American Association for Computational Linguistics (HLT-NAACL)*, pages 158–165, Edmonton, Canada.

I. Dan Melamed. 2004. Statistical machine translation by parsing. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, Barcelona, Spain.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, July.

Franz Josef Och, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the 4nd Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 20–28, College Park, Maryland.

G. Satta. 1992. Recognition of linear context-free rewriting systems. In *Proc. of the 30th ACL*, Newark, Delaware.

E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, and R. C. Carrasco. 2005. Probabilistic finite-state machines – Part I. *IEEE Trans. on Pattern analysis and Machine Intelligence*. To appear.

C. S. Wetherell. 1980. Probabilistic languages: A review and some open questions. *Computing Surveys*, 12(4):361–379.

Dekai Wu and Hongsing Wong. 1998. Machine translation with a stochastic grammatical channel. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL)*, Montreal, Canada, July.

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

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 531–538, Toulouse, July.