Submission to ICGI-2000

Computational complexity of problems on probabilistic grammars and transducers.

F. Casacuberta [*]
Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
46071 Valencia
(SPAIN)
fcn@iti.upv.es

C. de la Higuera
EURISE
Faculté des Sciences et Techniques,
Université de Saint Etienne - Jean Monnet
42023 Saint Etienne
(FRANCE)
cdlh@univ-st-etienne.fr

Abstract

Determinism plays an important role in grammatical inference. However, in practice, ambiguous grammars (and non determinism grammars in particular) are more used than determinism grammars. Computing the probability of parsing a given string or its most probable parse with *Stochastic Regular Grammars* can be performed in linear time. However, the problem of finding the most probable string has yet not given any satisfactory answer. In this paper we prove that the problem is **NP**-hard and does not allow for a Polynomial Time Approximation Scheme. The result extends to *Stochastic Regular Syntax-Directed Translation Schemes*.

Keywords: Syntactic pattern recognition, computational complexity, stochastic languages, stochastic translations.

Category of paper: REGULAR PAPER

All appropriate clearances for the publication of this paper have been obtained, and if accepted the author will prepare the final manuscript in time for inclusion in the Conference Proceedings and will present the paper at the Conference.

[*] Work partially supported by the Spanish CICYT under grant TIC95-0884-C04-01 and TIC97-0745-C02-02

1 Introduction

As the problem of not having negative evidence arises in practice when wishing to learn grammars, different options as how to deal with the issue have been proposed. Restricted classes of deterministic finite-state automata can be identified [**?**, **?**] heuristics have been proposed [**?**] and used for practical problems in speech recognition or pattern recognition [15], and stochastic inference has been proposed as a mean to deal with the problem [**?**, 18, 19].

Stochastic grammars and automata have been used for some time in the context of speech recognition [17, 16]. Algorithms that (heuristically) learn a context-free grammar have been proposed (for a recent survey see [**?**], and other algorithms (namely the forward-backward algorithm for hidden Markov models, close to stochastic finite automata or the inside-outside algorithm for stochastic context-free grammars) that compute probabilities for the rules have been realised [17, **?**]. But in the general framework of grammatical inference it is important to search for algorithms that not only perform well in practice, but that provably converge to the optimal solution, using only a polynomial amount of time. For the case of stochastic finite automata the problem has been dealt with by different authors: in [18] stochastic deterministic finite automata are learnt through Bayes minimisation, in [**?**], through state merging techniques common to classical algorithms for the deterministic finite-state automaton inference problem. Along the same line in [19] acyclic stochastic deterministic finite automata are learnt, proving furthermore that under certain restrictions the inferred automaton is probably approximately correct. Work in the direction of learning this sort of object has been followed these last years, with new algorithms proposed in [20, 22]. In a general sense the models that have been inferred are always deterministic. It is not obvious why this should be so as non-deterministic stochastic automata are strictly more powerful than their deterministic counter parts. They can also be of a smaller size and thus be more understandable. One reason may be that in the normal (non-stochastic) paradigm, it can be proved that non deterministic machines can not be identified in polynomial time [21]. In this work we point out that the difference between deterministic and non-deterministic stochastic automata (or regular grammars) is also that some reasonably easy problems in the deterministic case become intractable in the non deterministic case.

An appealing feature of Stochastic Regular Grammars is the existence of efficient algorithms for parsing. The probability of generating a given string by a Stochastic Regular Grammar can be computed in linear time with the length of the string. The same holds for the search of the derivation with the highest probability.

In spite of the existence of polynomial algorithms for dealing with some problems that involve Stochastic Regular Grammars, there is another important problem which does not have an efficient solution. This is *to find the most probable string* that can be generated by a Stochastic Regular Grammar.

Other useful models which are closely related to Stochastic Regular Grammars are the *Stochastic Regular Syntax-Directed Translation Schemes* [6, 9, 12]. Stochastic Grammars are adequate models for classification tasks; however, there are many practical situations which do not fit well within the classification framework but can be properly tackled through formal translation [10]. For translation, efficient (linear) algorithms are only known for the computation of the highest probability translation form [1]. In this framework, given an input string, the goal is to find its most probable translation. However, there is no efficient solution for this problem.

Under the Complexity Theory framework [7], we report some results about the difficulty of different computations regarding probabilistic finite state machines.

2 The Most Probable String Problem

The following definition is classical [9].

Definition 1: A *Stochastic Regular Grammar* (*SRG*) G is a tuple $< N, \Sigma, R, S, P >$, where N is a finite set of non-terminal symbols; Σ is a finite set of terminal symbols; R is a set of rules of the form $A \rightarrow aB$ or $A \rightarrow a$ for $A, B \in N$ and $a \in \Sigma$ (for simplicity, empty rules are not allowed); S is the starting symbol and $P : R \rightarrow Q^+$ (the set of the positive rational numbers) is a function such that

$$\sum_{a \in \Sigma, B \in N : (A \rightarrow aB) \in R} P(A \rightarrow aB) + \sum_{a \in \Sigma : (A \rightarrow a) \in R} P(A \rightarrow a) = 1 \;\; \forall A \in N$$

Stochastic Grammars are probabilistic generators of languages; therefore, the concept of probability that a string is generated by a *SRG* can be defined.

Definition 2: Given $w \in \Sigma^\star$ (the set of finite-length strings over Σ), the *probability that a SRG G generates* w is defined as:

$$pr_G(w) = \sum_{\forall d(w)} pr_G(d(w))$$

where $d(w)$ is a complete *derivation* of w in G of the form:

$$S \rightarrow w_1 A_1 \rightarrow w_1 w_2 A_2 \rightarrow ... \rightarrow w_1 w_2 ... w_{|w|-1} A_{|w|-1} \rightarrow w_1 w_2 ... w_{|w|-1} w_{|w|} = w$$

and

$$pr_G(d(w)) = P(S \rightarrow w_1 A_1) P(A_1 \rightarrow w_2 A_2) ... P(A_{|w|-1} \rightarrow w_{|w|})$$

Some important problems arise with these definitions. Namely the computation for a given string of its probability (*PS*) or of its most probable derivation (*MPDS*), the computation of the most probable derivation (*MPD*) and the computation of the most probable string (*MPS*). The PS, MPDS and MPD problems have been widely addressed. The PS and MPDS are classical parsing problems, and can be solved in time $O(|w||N|^2)$ [14]. The MPD problem can also be dealt with using Dijkstra's algorithm [3] to compute the shortest path in a weighted path and requires no more than $O(|N|^2)$ time. The MPS problem, although straightforward, has not been dealt with. Let us define the associated decision problem as follows:

Problem *Most Probable String* (*MPS*).
Instance A *SRG* G, and $p \in Q^+$.
Question Is there a string $x \in \Sigma^\star$ with $|x| \leq |N|$, such that $pr_G(x) \geq p$?

A more restricted problem is the following :

Problem *Restricted Most Probable String* (*RMPS*).
Instance A *SRG* G, $d \in \mathcal{N}$ (the set of natural numbers), $d \leq |N|$, and $p \in \mathcal{Q}^+$.
Question Is there a string $x \in \Sigma^d$ such that $pr_G(x) \geq p$?

RMPS is not just a special case of *MPS*. We will prove that both *MPS* and *RMPS* are **NP**-hard. As the probability of any string can be computed in polynomial time[*], both *MPS* and *RMPS* are in **NP**. We prove that *MPS* and *RMPS* are **NP**-hard by reduction from the "*Satisfiability*" problem (*SAT*) [7]. The proof relies on a technical encoding of a set of clauses:

Given an instance of SAT: 1) a collection $v_1, ..., v_n$ of n boolean variables and 2) a collection $c_1, ..., c_k$ of k clauses over the n variables, consider the following *SRG* $G = (N, \Sigma, R, S, P)$:

- $\Sigma = \{f, t, \$\, \#\}$
- For $1 \leq j \leq k$,

[*] This can be done in $O(|w||N|^2)$

- $A_0^j \in N$
- the rule $S \to \$A_0^j$ is in R with probability $1/k$ and rules $B_n^j \to \$$ and $A_n^j \to \#$ are in R with an associated probability 1.
- for $1 \leq i \leq n$ with an associated probability 1/2:
  - $A_i^j, B_i^j \in N$
  - the rules $B_{i-1}^j \to tB_i^j$ and $B_{i-1}^j \to fB_i^j$ are in R.
  - if v_i appears as a positive literal in c_j then the rules $A_{i-1}^j \to fA_i^j$ and $A_{i-1}^j \to tB_i^j$ are in R.
  - if v_i appears as a negative literal in c_j then the rules $A_{i-1}^j \to tA_i^j$ and $A_{i-1}^j \to fB_i^j$ are in R.
  - if v_i does not appear in c_j then the rules $A_{i-1}^j \to tA_i^j$ and $A_{i-1}^j \to fA_i^j$ are in R.

  Each of these rules have an associated probability of 1/2.

- For *RMPS* fix $d = n + 2$.

To illustrate this construction, consider an example where one of the clauses is $c_j = x_2 \vee \bar{x}_3 \vee x_5$ with $n = 6$. Then the corresponding part of the automaton[†] for this clause is shown in the Figure 1

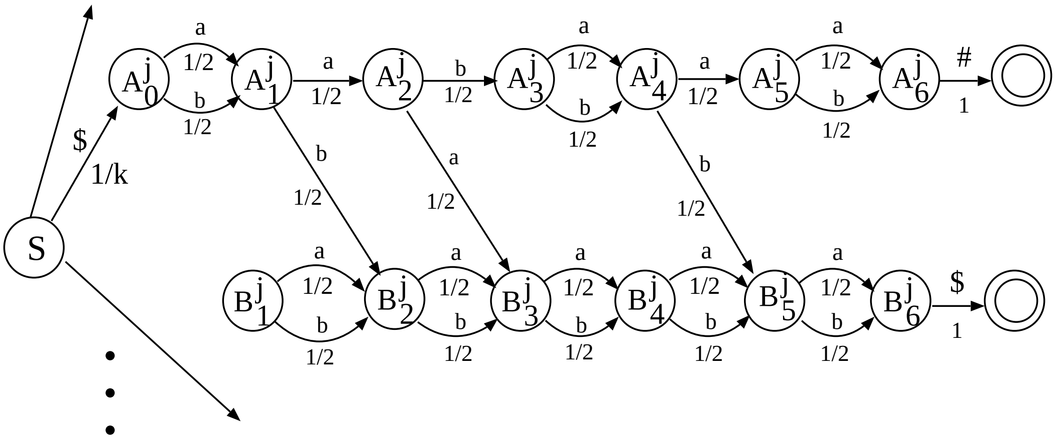

Figure 1: Part of the *SRG* corresponding to clause $c_j = x_2 \vee \bar{x}_3 \vee x_5$ with $n = 6$.

This SRG must verify that if a clause is evaluated to *true* by the assignment of values to the boolean variables, then a complete derivation has to exist and vice-versa. On the other hand, if a clause is evaluated to *false*, no such derivation associated to the clause has to exist in the SRG nor vice-versa.

Theorem 1: *RMPS* and *MPS* are **NP**-complete.

Proof of Theorem 1:

From the above construction, if the clause is satisfied for some solution $(x_1, ..., x_n)$ the corresponding string $\$X_1...X_n\$$ (X_i is t if x_i is $true$, and X_i is f if x_i is $false$) will have probability $1/(k\,2^n)$ for each derivation linked to the clause. Note that the construction is in $O(k\,n)$. The string length is $n + 2 \leq |N| = k \cdot (2n + 1)$.

Fix[‡] p to $1/2^n$. Let m be a solution of *SAT*; it can be considered as a string in $\{f, t\}^n$, hence the corresponding $\$m\$$ is a string generated by G with k derivations all of probability $1/(k\,2^n)$; so the probability of $\$m\$$ by G is $1/2^n$. On the other hand, if the instance of *SAT* does not admit any

[†] A *SRG* can be interpreted by its associated graph. Notice that some states (A_6^j and B_1^j) are useless). The fact that the grammar is not in proper normal form is irrelevant.

[‡] Encoding of p only requires n bits

solution, then as the only strings that have non null probability for the associated grammar are of length $n + 2\,(= d)$, and at least one clause is not satisfied (for example if the clause j is not satisfied, the corresponding derivation ends in A^j_n), then no string has probability $1/2^n$ $\bigtriangledown$.

Consequently the corresponding optimization problems (finding the most probable string) are **NP**-hard. More can be said about the **NP**-optimization problem:

Problem *Maximum Probability of a String* (*MaxPS*).
Instance A *SRG* G , and $p \in Q^+$.
Solution A string $x \in \Sigma^\star$
Measure $pr_G(x)$.

By reduction from *Maximum Satisfiability* (*Max-SAT*) [8, 11],

Theorem 2: *MaxPS* is **APX**-hard.

Maximum Satisfiability is the *NP*-optimization problem corresponding to *SAT*. It concerns finding a subset of clauses such that there is a truth assignment satisfying each clause in the subset. The associated measure is just the number of clauses.

The problem is **APX**-complete, i.e. it is complete for the class **APX**. Being **APX**-complete implies that you can not do better than a constant approximation (a bound of the constant approximation is proposed by Goemans and Williamson [8]) and that no *PTAS* (polynomial time approximation scheme) is feasible.

Proof of Theorem 2:

The proof is straight forward and involves the same construction as for the **NP**-hardness of *MPS*:

Given an instance I of *Max-SAT*, and a rational ϵ, construct an instance $f(I, \epsilon)$ of *MaxPS* as in the proof of theorem 1. Now given a string x on the input alphabet of the associated *SRG* $f(I, \epsilon)$, the following holds:

$$pr_{f(I,\epsilon)}(x) = \frac{c}{k\,2^n} \Rightarrow c = g(I, x, \epsilon) \text{ clauses of } I \text{ can be satisfied.}$$

Finally we have, for any instance I of *Max-SAT*, any rational ϵ and any string x solution to $f(I, \epsilon)$:

$$\frac{opt(f(I, \epsilon))}{m(f(I, \epsilon), x)} = \frac{opt(I)}{m(x, g(I, x, \epsilon))}$$

where *opt* denotes the optimal result (maximum number of satisfied clauses or maximum probability) and m is the measure function (number of actual satisfied clauses for a given assignment and probability of a given string). It follows that with ϵ playing a dummy part the reduction inequation can be obtained [4]:

$$\frac{opt(f(I, \epsilon))}{m(f(I, \epsilon), x)} \le \epsilon \Rightarrow \frac{opt(I)}{m(x, g(I, x, \epsilon))} \le \epsilon$$

All these constructions are polynomial. $\bigtriangledown$

3 Stochastic Regular Syntax-Directed Translation Scheme

The last problem deals with the search of an optimal translation of a given input string according to a translation scheme [9].

Definition 3: A *Stochastic Regular Syntax-Directed Translation Scheme* (*SRT*) E is a tuple $< N, \Sigma, \Delta, R, S, P >$, where N and S are defined as in *SRG*s, Σ is a finite set of *input terminal symbols*, Δ is a finite set of *output terminal symbols* ($\Sigma \cap \Delta = \emptyset$); R is a set of rules of the form $A \rightarrow aB, \omega B$ or $A \rightarrow a, \omega$ for $A, B \in N$, $a \in \Sigma$, $\omega \in \Delta^\star$ and and $P : R \rightarrow Q^+$ is a function such that

$$\sum_{\forall a \in \Sigma, \omega \in \Delta^\star, B \in N : A \rightarrow aB, \omega B \in R} P(A \rightarrow aB, \omega B) + \sum_{\forall a \in \Sigma, \omega \in \Delta^\star : A \rightarrow aB, \omega B} P(A \rightarrow a, \omega) \; \forall A \in N$$

For simplicity, empty input rules ($A \rightarrow \lambda B, wB$ or $A \rightarrow \lambda, w$ where λ is the empty string) are not allowed.

*SRG*s and *SRT*s are closely related and given a *SRT* E, the probability of a translation pair $(x, y) \in \Sigma^\star \times \Delta^\star$, $pr_E(x, y)$ is defined in a way similar to that for *SRG*s:

Definition 4: The *probability of a translation pair* $(x, y) \in \Sigma^\star \times \Delta^\star$ according to the scheme E is defined as:

$$pr_E(x, y) = \sum_{\forall t(x,y)} pr_E(t(x, y))$$

where $t(x, y)$ is a *translation form* of (x, y) in E:

$$(S, S) \rightarrow (x_1 A_1, y_1 A_1) \rightarrow (x_1 x_2 A_2, y_1 y_2 A_2) \rightarrow ... \rightarrow (x, y)$$

and the corresponding probability of the translation form is:

$$pr(t(x, y)) = P(S \rightarrow x_1 A_1, y_1 A_1) P(A_1 \rightarrow x_2 A_2, y_2 A_2) .... P(A_{|x|-1} \rightarrow x_{|x|}, y_{|x|})$$

The following example is presented to illustrate the above definitions.

Example 1. $N = \{S, A, B\}$, $\Sigma = \{0, 1\}$, $\Delta = \{a, b\}$ and the rules of the Table 1. The input string 010 has two possible translations: $abbbba$ and $aaaaa$. The first one can be obtained as $S \rightarrow 0B, abB \rightarrow 01A, abbbbA \rightarrow 010, abbbba$ with probability 1/25, and the second one with probability 6/245 as $S \rightarrow 0A, aA \rightarrow 01A, aaaaA \rightarrow 010, aaaaa$ or with probability 9/175 as $S \rightarrow 0A, aA \rightarrow 01B, aaaB \rightarrow 010, aaaaa$. Therefore, 010 can be translated into $abbbbba$ with probability 1/25, or into $aaaaa$ with probability 6/245 + 9/175 =93/1225.

Table 1: Set of rules and probabilities corresponding to *SRT* of Example 2.

Rules (R)	Probabilities (P)
$S \rightarrow 0A, aA$	3/10
$S \rightarrow 0B, abB$	7/10
$A \rightarrow 1B, aaB$	2/7
$A \rightarrow 1A, aaaA$	4/7
$A \rightarrow 0, a$	1/7
$B \rightarrow 1A, bbbA$	2/5
$B \rightarrow 0, aa$	3/5

An interesting question is thus the one of computing the most probable translation of some input string. Formally:

Problem *Most Probable Translation* (*MPT*).
Instance A *SRT* E , $x \in \Sigma^\star$ and $p \in Q^+$.
Question Is there an output string $y \in \Delta^\star$, $|y| \leq |N| \cdot l_{max}$ (l_{max} is the maximum length of the output string in a rule) such that $pr_E(x, y) \geq p$?

In Example 1, the second translation ($aaaaa$) has the highest probability, therefore it is the most probable translation of 010.

If the translation defined by E from $\Sigma^\star$ to $\Delta^\star$ is not ambiguous (E defines a function from $\Sigma^\star$ to $\Delta^\star$), there is an efficient algorithm that computes an answer to the *MPT* problem in linear time. Basically, this algorithm performs a parsing of the input with the input grammar.

The *MPT* problem can be reduced from *RMPS* as follows: given a *SRG* $G =< N, \Sigma, R, S, P >$, an integer n and a rational p, construct: a *SRT* $E =< N', \Sigma', \Delta, R', S', P' >$ with

- $N' = N$,
- $\Delta = \Sigma$,
- $\Sigma' = \{\$\}$,
- for every rule $A \rightarrow aB \in R$, a rule $A \rightarrow \$B, aB$ is in R' with $P'(A \rightarrow \$B, aB) = P(A \rightarrow aB)$
- for every rule $A \rightarrow a \in R$, a rule $A \rightarrow \$, a$ is in R' with $P'(A \rightarrow \$, a) = P(A \rightarrow a)$
- an input string $\$^n$ ($n \leq |N|$)
- a rational p

Theorem 3: *MPT* is **NP**-complete.

> **Proof of the Theorem 3:** From the above reduction, it follows that: 1) the construction is polynomial; and 2) $\$^n$ has an output string $y \in \Delta^\star$ such that $pr_G(y) \geq p$ if and only if $pr_E(\$^n, y) \geq p$. The length of y
>
> $\bigtriangledown$

And the associated optimization problem of computing the most probable translation is **NP**-hard. Without proof, (it follows from the previous different results and proofs) for the associated **NP** optimization problem (*MaxPT*) we give a final result:

Theorem 4: *MaxPT* is **APX**-hard.

4 Conclusions

In this paper we have presented computational complexity results regarding parsing problems for Stochastic Regular Grammars and Stochastic Regular Syntax-Directed Translation Schemes. In particular, the problems of searching for the most probable string in a *SRG* and of searching for the most probable translation of an input string given a *SRT* are **NP**-hard problems and the associated optimization problems do not admit polynomial approximation schemes. Future work can be conducted in the following direction: we have proved that both *NP*-optimization problems are **APX**-hard. Do they belong to **APX**? Such a result would require a polynomial time algorithm that always meets a given bound.

References

[1] D. Angluin, Inference of reversible languages. Journal of the ACM, Vol. 29(3), 741-765, 1982.

[2] R. Carrasco and J. Oncina, Learning stochastic regular grammars by means of a state merging method, in *Grammatical Inference and Applications*. Proceedings of ICGI '94, Lecture Notes in Artificial Intelligence 862, Springer Verlag ed., 139-150, 1994.

[3] Carrasco, J. Oncina, Learning deterministic regular grammars from stochastic samples in polynomial time. Informatique Théorique et Applications, Vol. 33(1), 1-19, 1999.

[4] F.Casacuberta, Maximum mutual information and conditional maximum likelihood estimations of stochastic syntax-directed translation schemes, in: L. Miclet and C. de la Higuera (eds), *Grammatical Inference: Learning Syntax from Sentences*. Lecture Notes in Artificial Intelligence, Vol 1147, 282-291, Springer-Verlag, 1996.

[5] F.Casacuberta, Growth transformations for probabilistic functions of stochastic grammars. International Journal on Pattern Recognition and Artificial Intelligence. Vol. 10, 183-201, 1996.

[6] T. Cormen, Ch. Leiserson and R. Rivest, Introduction to algorithms. The MIT Press, 1990.

[7] P. Crescenzi and V. Kann, A compendium of **NP** optimization problems, http://www.nada.kth.se/ viggo/problemlist/compendium.html (1995).

[8] K.S. Fu and T.L.Booth, Grammatical inference: introduction and survey. Part I and II, IEEE Transactions on System Man and Cybernetics, Vol. 5, 59-72/409-23, 1985.

[9] K.S. Fu, Syntactic pattern recognition and applications. Prentice-Hall, Englewood Cliffs, NJ. 1982.

[10] P. García and E. Vidal, Inference of K-testable languages in the strict sense and applications to syntactic pattern recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 12(9). pp. 920-925. 1990.

[11] M.R. Garey and D.S. Johnson, Computers and intractability: a guide to the theory of NP-completeness W.H. Freeman, San Francisco, 1979.

[12] M.X. Goemans and D. O. Williamson, 878-approximation algorithms for MAX-CUT and MAX-2SAT. Proc. Twenty sixth Ann. ACM Symposium on Th. of Comp., 422-431, 1994.

[13] R. González and M. Thomason, Syntactic pattern recognition: an introduction. Addison-Wesley, Reading, MA 1978.

[14] K. Lari, and S. Young, Applications of stocashtic context-free grammars. Computer Speech and Language. Vol. 5. 237-257. 1991.

[15] S. Lucas, E. Vidal, A. Amiri, S. Hanlon and J-C.Amengual, A comparison of syntactic and statistical techniques for off-line OCR. Proceedings of the International Colloquium on Grammatical Inference ICGI-94 (pp. 168-179). Lecture Notes in Artificial Intelligence 862, Springer-Verlag, 1994.

[16] H. Ney, Stochastic grammars and Pattern Recognition, in Speech Recognition and Understanding. edited by P. Laface and R. de Mori, Springer-Verlag, 45-360, 1995.

[17] C. de la Higuera, Characteristic sets for grammatical inference Machine Learning, 27 pp. 1-14, 1997

[18] J. Oncina, P. García and E. Vidal, Learning subsequential transducers for pattern recognition tasks, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 15, 448-458, 1993.

[19] C.H. Papadimitriou and M. Yannakakis, Optimisation approximation and complexity classes. Journal Computing System Science, Vol. 43, 425-440, 1991.

[20] L. Rabiner and B.H. Juang, Fundamentals of Speech Recognition. Prentice-Hall, 1993.

[21] D. Ron, Y. Singer and N. Tishby, On the Learnability and Usage of Acyclic Probabilistic Finite Automata, Proceedings of COLT 1995 , 31-40, 1995.

[22] H. Rulot and E. Vidal, Modelling (sub)string-length-based constraints through grammatical inference methods. Devijver and Kittler eds. Sringer-Verlag 1987.

[23] Y. Sakakibara, Recent Advances of Grammatical Inference. Theoretical Computer Science Vol. 185, 15-45, 1997.

[24] A. Stolcke and S. Omohundro, Inducing Probabilistic Grammars by Bayesian Model Merging, in Grammatical Inference and Applications. Proceedings of ICGI '94, Lecture Notes in Artificial Intelligence 862, Springer Verlag ed., 106-118, 1994.

[25] F. Thollard and P. Dupont and C. de la Higuera, Probabilistic DFA Inference using Kullback-Leibler Divergence and Minimality. ICML2000 (International Colloquium on Machine Learning), Stanford, 2000.

[26] E.Vidal, F.Casacuberta and P.García, Syntactic Learning Techniques for Language Modeling and Acoustic-Phonetic Decoding, in: A. Rubio (ed.) *New Advances and Trends in Speech Recognition and Coding* Chap. 27, NATO-ASI Series Springer-Verlag, 1995.

[27] M. Young-Lai and F.W. Tompa, Stochastic Grammatical Inference of Text Database Structure, to appear in Machine Learning, 2000.