Effective Structural Inference for Large XML Documents

Jason Sankey **Raymond K. Wong**
School of Computer Science & Engineering
University of New South Wales
Sydney 2052, Australia

Abstract

This paper investigates methods to automatically infer structural information from large XML documents. Using XML as a reference format, we approach the schema generation problem by application of inductive inference theory. In doing so, we review and extend results relating to the search spaces of grammatical inferences for large data set. We evaluate the result of an inference process using the concept of Minimum Message Length. Comprehensive experimentation reveals our new hybrid method to be the most effective for large documents. Finally tractability issues, including scalability analysis, are discussed.

1 Introduction

Given the recent emergence of XML, there are many problems that must be solved to facilitate its most effective use. Amongst the most important of these involves addressing the differences between the new, loosely formatted data and traditional, structured data. Clearly, the ability to infer the structure of XML documents would be a very powerful tool for bridging the gap. Given such a method of inference, XML information may be handled in more effective ways without loosing the advantage of flexibility. There are still many possible approaches to the problem, and correspondingly many suitable outcomes. For this reason it is best to firstly derive a suitable measure for the quality of an inferred DTD. As will be seen in section 3.1, determining the relative utility of content models is a task in itself. This paper extends the hybrid method from our previous work (Sankey & Wong 2001) to automatically infer structural information from large XML documents. Comprehensive experimentation reveals the proposed method to be the most effective for large documents.

The paper is presented in the following fashion. Section 2 provides details of previous work in this and related fields. Section 3 provides an overview of our solution method, followed by details of the inference algorithms in section 4. Section 5 includes comprehensive testing for large models and section 6 discusses the tractability considerations, followed by conclusions in section 7.

2 Previous Work

The inference of structure in XML information is a relatively new area of research. However, there are several closely related topics that have been studied for a longer period. Many of these topics fall into the general field of Inductive Inference, more specifically the sub-field of Grammatical Inference. This sub-field is concerned with the theory and methods for learning grammars from example data. For further details concerning the field of Grammatical Inference the reader is referred to the surveys of Pitt (Pitt 1989) and Sakakibara (Sakakibara 1997). In addition, there has also been prior research into automatic recognition of document structure. Earlier attempts by (Chen 1991), (Fankhauser & Yu 1994) and (Shafer 1995) in similar problem spaces all use solutions based on heuristic methods. In each case, the generalisation step involves searching for similar patterns in the data and combining the corresponding structural information. Although these techniques may work well in some cases, their applicability is restricted by a lack of generality. The approaches of (Ahonen 1996) and (Young-Lai 1996) are more powerful in concept. In both of these works, methods derived from theoretical grammatical inference are applied to the problem of inferring DTD content models. The first known application of such theory to this problem, in (Ahonen 1996), makes use of a characterising method to infer a subset of the regular language class. The alternative solution in (Young-Lai 1996) makes use of an adapted stochastic method. In both cases, the results are post-processed to produce more desirable content models. Unfortunately, neither paper investigates or compares other methods. It is partly for this reason that these methods have been included in this study for comparisons. The more recent paper of Garofalakis et al (Garofalakis et al. 2000) is very similar to this work in terms of motivation and the application of information theory. However, their inference algorithms are based upon direct generalisation and factoring of regular expressions, with information theoretic principles used to choose a final result from a pool of candidates. In contrast, we propose a hybrid method which employs various principles throughout the inference process, with the aim of producing a more general method.

3 Overview of Solution

The major grammatical inference methods fall into a few general categories. One of these categories includes a family of algorithms known as state merging methods. A state merging method typically begins by constructing what is known as a **Prefix Tree Automaton** (PTA) from the positive examples of the language to be inferred. If we let the set of positive examples be $R+$, then the prefix tree for $R+$ ($PTA(R+)$) may be constructed as follows. We begin with an automaton that simply accepts the first string in $R+$. Then we iterate over the rest of the strings in $R+$ and for each one follow transitions in the automaton for as many symbols as possible. When a symbol is found that does not match a valid transition, the PTA is augmented with a new path to accept the rest of the string. In particular, a **Probabilistic Finite State Automaton** (PFSA) is merely an automaton with probabilities associated with each transition and final state. This is important both for some of the inference methods and for evaluating the quality of solutions.

3.1 Evaluating a Solution

To measure the quality of the inferred DTD, we use the concept of Minimum Message Length (MML) (Georgeff & Wallace 1984). In particular, we adapt the formula developed for PFSA (Raman 1997) as below:

$$\begin{aligned} MML(A) &= \sum_{j=1}^{N} \left\{ \log_2 \frac{(t_j - 1)!}{(m_j - 1)! \prod_{i=1}^{m_j} (n_{ij} - 1)!} \right\} \\ &\quad + M(\log_2 V + 1) + M' \log_2 N \\ &\quad - \log_2 (N-1)! \end{aligned}$$

where:

- N is the number of states in the PFSA
- V is the cardinality of the alphabet plus one
- t_j is the number of times the jth state is visited
- m_j is the number of arcs from the jth state (plus one for final states)
- m'_j is the number of arcs from the jth state (no change for final states)
- n_{ij} is the frequency of the ith arc from the jth state
- M is the sum of all m_j values and
- M' is the sum of all m'_j values

3.2 Implementation

One of the goals of this work was to both produce new inference methods and make comprehensive comparisons with existing techniques. This entailed a significant amount of implementation that consists of several modules to perform stages of the inference process. The most important stage involves PTA generalisation using the inference methods. These methods fall into two broad categories, which are labelled generalisation and optimisation in the implementation. The first of these consists of the Merge methods with its pseudo-code shown in algorithm 1.

Algorithm 1 GeneralisePTA

```
Input: A PTA $A$ to be generalised, a merge criterion $criterion$
Output: The generalised form of $A$
Method:
 1. repeat
 2.   for all pairs $(s_1, s_2)$ of states in $A$ do
 3.     if $criterion(s_1, s_2)$ then
 4.       $A.merge(s_1, s_2)$
 5.       $criterion.forcedMerges(A)$
 6.       if $criterion.determinise()$ then
 7.         determinise($A$)
 8.       end if
 9.     end if
10.   end for
11. until no more merges are possible
12. if not $criterion.determinise()$ then
13.   determinise($A$)
14. end if
15. return $A$
```

Here the merge criterion determines the actual behaviour of the inference procedure. For each method belonging to the merge family, a merge criterion class is derived from a base interface. The merge criterion is allowed to make forced merges after an initial merge is decided (see line 5), which may be necessary to fit the semantics of a method, or may be more efficient. Also, merge criteria may decide if the PFSA is determinised after every merge (lines 6–8) or only at the end of the inference process (lines 12–14). Determinisation itself is performed by merging of states, as opposed to the traditional algorithms. The alternative inference methods all apply optimisation techniques to try and minimise the MML of the PTA. These algorithms vary significantly in implementation, ruling out the possibility of building them around the same core algorithm.

4 Inference Algorithms

4.1 Reference Methods

Several previously applied methods were implemented to evaluate their relative performance. Two such algorithms are those applied by Ahonen in (Ahonen 1996), the first known paper to address DTD generation using tradition grammatical inference methods. These methods are theoretically appealing, as they guarantee to infer languages falling within certain language classes. These classes are termed k-contextual and (k, h)-contextual, so named as they assume the structure to be inferred to have limited context. It is not clear whether this assumption is valid in practice, however. Another method applied to DTD generation ((Young-Lai 1996)), is derived from more recent work in grammatical inference. The base algorithm is known as Alergia, introduced in (Carrasco & Oncina 1994b). The criterion for state equivalence in Alergia is based upon observed frequencies of transitions and finalities of a pair of states. As with the methods of Ahonen, Alergia has strong theoretical appeal. Again, though, we are interested in practical performance. Further to the methods described above, we devised two basic optimisation strategies against which to benchmark the results of our main algorithm. The first of these, termed the Greedy method, is a straightforward steepest-descent algorithm which employs incremental MML calculation to optimise a PTA. Along with this a weighted stochastic hill-climbing method was implemented, which also used incremental MML calculation. These two methods illustrate the need for more sophisticated optimisation algorithms.

4.2 The sk-strings Method

The sk-strings method actually consists of a family of algorithms, described in (Raman & Patrick 1997) and in more detail in (Raman 1997), of which five were implemented. The basis of these algorithms is an extension upon the k-tails heuristic of Biermann and Feldman (Biermann & Feldman 1972), which in turn is a relaxed variant of the Nerode equivalence relation. Under the Nerode relation, a pair of states are equivalent if they are indistinguishable in the strings that may be accepted following them. The k-tails criterion relaxes this to only considering these strings (tails) up to a given length (k.) The sk-strings method is extended using stochastic automata, and considers only the top s percent of the most probable k-strings. The k-strings differ from k-tails in that they are not required to end at a final state, unless they have length less than k. The probability of a k-string is calculated by taking the product of the probabilities of the transitions exercised by that k-string.

4.3 Ant Colony Optimisation

The Ant Colony Optimisation (ACO) meta-heuristic is a relatively new optimisation technique. First described in (Dorigo et al. 1991), the method uses a positive feedback technique for searching in a similar manner to actual ants. In biological experiments it was revealed that insect ants cooperated in finding shortest paths by leaving pheromone trails as they walked. An ant traveling back and forth along a short path increases the pheromone level on that path more rapidly than an ant using a longer path, thus influencing more ants to take the shorter route. The effect is then self-reinforcing until eventually all ants will choose the shorter path. ACO algorithms mimic this technique using artificial ants to search out good solutions to optimisation problems.

The ACO heuristic operates over several iterations to allow the positive feedback of the pheromones to take effect. In each iteration, the artificial ants navigate the search space using only a simple heuristic, but as they move they leave pheromones on the trail they follow. In some variants, including the one used in this work, the pheromone placement is delayed until the end of an iteration, when all ants have completed a full walk. At this point the amount of pheromone assigned to each ant is weighted with respect to the quality of the solution it found. Thus moves involved in higher quality solutions are more likely to be chosen by ants in future iterations. Although ants acting by themselves are only capable of finding relatively poor solutions, working in cooperation they may approach higher quality ones. After a certain number of iterations without improvement to the best solution found the algorithm terminates.

4.4 The Proposed Hybrid Method

The sk-ANT Heuristic: The motivation for this new heuristic was to create a method that would be successful for a variety of input data sizes by combining the best features of both the sk-strings and ACO techniques. One consideration was to first run the sk-strings algorithms, and then use the results to seed the ACO optimisation. However, this approach suffers from several problems. Firstly, it is not practical to attempt all possible combinations of both algorithms. Thus we would be required to choose a limited number of models resulting from the sk-strings technique to seed the second stage of the process. The simplest way to achieve this would be to choose the best models, up to a reasonable number. These models will not necessarily lead to the best results, though, as they may have already been over-generalised. More importantly, by letting the sk-strings methods run to completion we would lose many of the advantageous aspects of the ACO method. Most notably, its willingness to explore a greater breadth of the search space would be missed.

The new method thus incorporates both the sk-strings and ACO heuristics at each step of the inference process. It is most easily described as a modified version of the ACO technique with the ants guided by an sk-strings criterion. The guiding is made progressively weaker as the model becomes smaller, to allow the advantages of the ACO method for smaller models to take effect. The key of this new method is a new algorithm for the ant move selection as shown in algorithm 2. In particular in line 4, a merge must pass the sk-strings criterion to be considered. The outer while loop on line 2 and if statement on line 11 combine to progressively weaken the sk-strings criterion when it has become too strict. Eventually the criterion will be weak enough to let all merges pass, and the algorithm will behave identically to the original version.

Algorithm 2 sk-antMoveSelector

Input: A set of all state pairs *merges*, an ant heuristic *heuristic*, an ant weighting function *weighting*, a pheromone table *pheremones* and an sk-strings criterion *skCriterion*.

Output: A state pair representing the chosen merge.

Method:

```
1.  $choices \leftarrow [\,]$
2.  while $choices.size() = 0$ do
3.      for $merge$ in $merges$ do
4.          if $skCriterion(merge)$ then
5.              $h \leftarrow heuristic(merge)$
6.              $p \leftarrow pheremones[merge]$
7.              $value \leftarrow weighting(h, p)$
8.              $choices.add((value, merge))$
9.          end if
10.     end for
11.     if $choices.size() = 0$ then
12.         $skCriterion.weaken()$
13.     end if
14. end while
15. return stochasticChoice($choices$)
```

5 Experimental Results

To fulfill our goal of comprehensive testing, we applied three sets of tests. The first two of these used generated data, which allowed systematic experimentation on widely varied input. The other test set consisted of a few models chosen from real data, to illustrate the nature of the models inferred by the sk-ANT method. In each test the algorithms were run with a range of input parameters, and the results from each run combined.

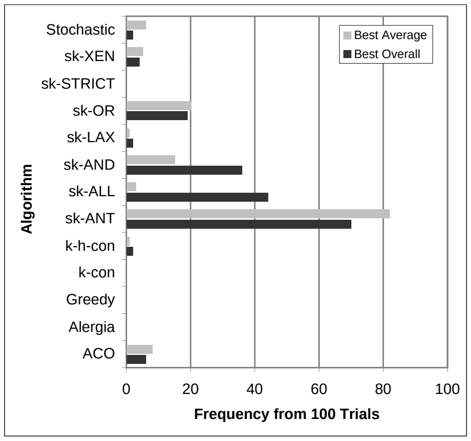

Figure 1: Success rates for large models

5.1 Inference of Large Models

The larger data set also consisted of 100 sample files generated from random PFSA. In this case the PFSA had a maximum of 20 states with an alphabet cardinality of 8. For each of these a total of 25 strings were generated giving an average PTA size of 143.38 states. On average the inferred models were larger than for the small data set, though they ranged from an average of 1.23 to 142.74 states for differing algorithms.

Figure 1 shows the success rates of each algorithm in inferring models with the lowest MML values. For each algorithm, two results have been shown. The first is the frequency of inferring the best model overall, by choosing the best of the algorithm's attempts. The second is the frequency of obtaining the best average performance, derived by averaging all of the algorithm's attempts before ranking. The best overall performance is most important, whilst the best average indicates stability across different input parameters. In particular, sk-ALL denotes the implementation of sk-strings with five variants from its family (refer to (Raman 1997) for details of different variants). The results show the poor performance of both the Stochastic and ACO methods, and the clear dominance of the sk-ANT heuristic. The poor performance of the ACO and Stochastic methods is likely due to the large search space. The heuristic guidance used in the sk-ANT method clearly overcomes this difficulty, producing the best results. Other poor algorithms are desirable due to their simplicity and efficiency, but are lacking in quality of solutions found.

The deviations from the best model were calculated for each algorithm, as presented in table 1. The numbers were derived from the difference between the MML values of the best model inferred by a given algorithm as compared with the best model overall. The hybrid sk-ANT technique again

Algorithm	Average Deviation (%)	Worst Deviation (%)
ACO	31.57	154.87
Alergia	32.44	86.58
Greedy	173.46	574.48
k-contextual	30.39	84.15
(k, h)-contextual	21.57	58.57
sk-ANT	2.81	28.52
sk-ALL	3.15	20.51
Stochastic	36.40	159.11

Table 1: Deviation from the best model inferred (large data set)

proved to be the best in terms of average deviation at 2.81%. Thus the newer method is preferable if only one choice is allowed. On the other hand, the sk-ALL heuristic was better in terms of worst-case performance, though the deviation is still rather high. Again, some applications may need to employ a combination of difference methods to achieve better worst-case results.

The experiments have revealed many interesting points. Firstly, the k-contextual, (k, h)-contextual and Alergia methods previously applied to this problem have been shown to perform poorly. Although the papers describing their use extend the algorithms and employ refining operations, it appears that other methods are a more appropriate starting point. We have also seen that the simple Greedy and Stochastic methods cannot match the performance of more complicated techniques. This highlights the difficulties inherent in the search space of grammatical inference. The sk-strings method developed in (Raman 1997) has proven to be much more effective, provided combined results from all of the heuristics are used. A major contribution to its success may lie in its use of statistical data in the inference process. The ACO algorithm's failure on large testing data led to a new method which we have named sk-ANT. This new hybrid algorithm proved to be the most effective by a considerable margin, and is thus the algorithm of choice. Where worst case guarantees are required, we recommend using combined results of both the sk-ANT and sk-ALL methods.

5.2 Real Data Set: Webster's p Element

The real data set we used was an extract of the digital form of the 1913 Webster Unabridged Dictionary. The dictionary format is almost compatible with SGML, and after some preprocessing we were able to extract the structural information in XML format. Only a small subset of the entire dictionary was used, due to its overwhelming size. We selected the paragraph element 'p', which is used to group together the information pertaining to each dictionary word, for the experiment. This particular element was chosen as it exhibits a rather complex and variable structure. This is in contrast to the structure

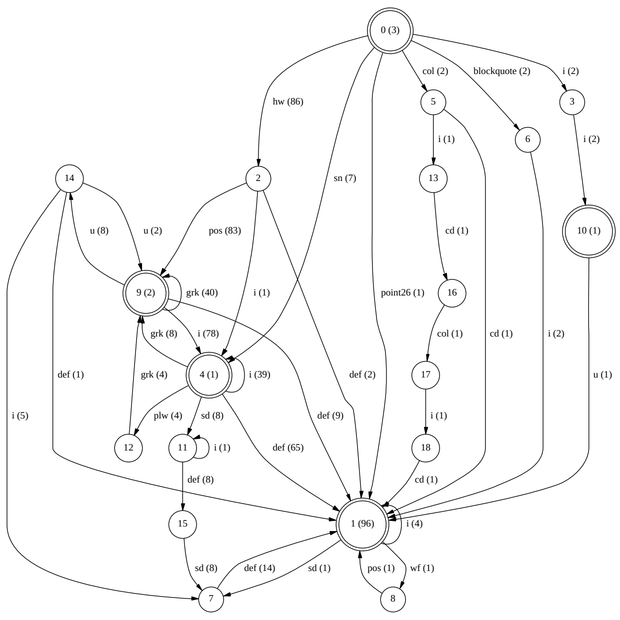

Figure 2: Best model for the 'p' element

of the other models chosen from real data sets, with the intent being to stretch the capabilities of the sk-ANT method.

The inferred model is shown in figure 2. Observe the dominant path through states 0, 2, 9, 4 and 1. It is most important that this path is preserved, with the rest of the structure accounting for exceptions and noise in the data. Such irregularity is a fact of life when dealing with semi-structured data such as XML.

6 Tractability Considerations

The goals of grammatical inference such as the structural inference presented in this paper require not only that our methods be effective, but also that they are useful in practice. For this reason, we investigate the complexity of our new algorithm of choice, namely sk-ANT. Coverage of the tractability of the other algorithms may be found in the original papers in which they are presented ((Ahonen 1996), (Carrasco & Oncina 1994b) and (Raman 1997).) In our testing, we found that the sk-ANT method was the most expensive in terms of running time. This did not provide an obstacle for the experimental data, but may become important when working with very large PTA (several thousand states) or under tight time constraints. Fortunately, in practice, a model for typical large XML documents would be normally fewer than one hundred states.

By analysing the sk-ANT algorithm, we found that the most important factor is the number of merges considered at each iteration of the inference process. At every step, each one of the possible state pairings is considered for merging. At a stage when

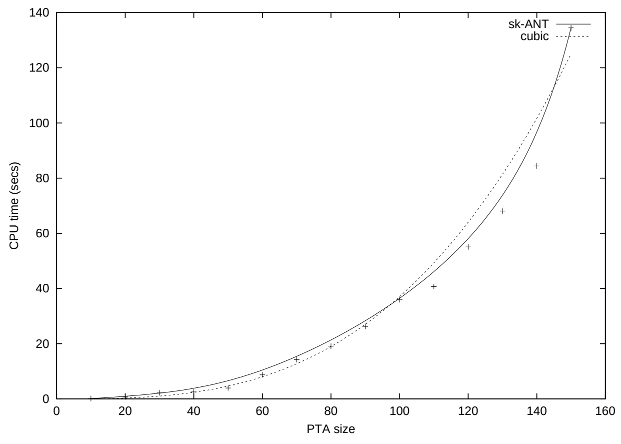

Figure 3: Empirical measure of sk-ANT complexity

the PFSA being inferred has r states, there are $\binom{r}{2}$ such merges. This may be expanded to give:

$$\frac{r(r-1)}{2}$$

merges considered. As the algorithm proceeds from the full number of states, say n, until there is just one state, the total number of merges considered is:

$$\sum_{r=n}^{2} \frac{r(r-1)}{2}$$

These merges are considered by all ants, introducing an extra constant factor. This factor does not influence the asymptotic complexity, however, which is clearly $O(n^3)$. Although not unmanageable, there are applications for which this complexity may be too great.

6.1 Measured Scalability

To complement the theoretical analysis, we also performed a simple empirical test to measure the performance of the algorithm in practice. The test was performed on PTA with sizes ranging from 10 to 150 states, with several of each size. Keeping in line with the experimental results shown earlier, we used the same range of parameters and averaged the results for each PTA size.

Figure 3 shows a graph of the obtained timing values. The points give the actual values for each PTA size, with a bezier approximation shown using a solid line. For comparison, we have also included a cubic function, shown with the dotted line. Clearly the analytical result corresponds well to the empirical measurement, with the cubic function proving to be a reasonable approximation of the points. Note that the individual CPU times shown are not of particular interest, as the primary goal of the implementation was not efficiency. Profiling has shown that there is scope for improvement, though of course the asymptotic growth will remain the dominant factor.

6.2 Discussion

The analytical and empirical analysis of the sk-ANT algorithm have both shown it to have asymptotic complexity of $O(n^3)$. In practice, this is quite acceptable for a wide range of applications. Where it is not acceptable, there are several alternatives. A simple one would be to employ one of the more efficient algorithms until the PFSA inferred reaches a small enough size to seed the sk-ANT method. A more complicated method may use a modified sk-ANT heuristic that employs approximations, and perhaps limits the size of the neighbourhood examined by each individual ant. Such a method may be able to reduce the complexity to $O(n^2)$, though it has not been thoroughly investigated. Cases where large amounts of data are involved can often be broken down into several sub-problems to make them manageable. For instance, in a database where documents are drawn from many sources, it may be appropriate to treat each data source separately. Indeed, this is a requirement for many applications. In other cases, it is feasible to take a sample of the data to use for inference, as the sk-ANT algorithm has been shown to perform well on sparse examples.

7 Conclusion

We have addressed the problem of structural inference for large XML documents. In doing so we began by motivating the research and reviewing the literature. The use of Minimum Message Length as a measure for the quality of inferred content models has been introduced, adapted from work in related fields. This measure has proven to be an appropriate and a vast improvement over previous subjective techniques. We have also presented the first wide spread comparison of different grammatical inference techniques. This involved the implementation of the existing Alergia, k-contextual, (k, h)-contextual and sk-strings methods, as well as the creation of new algorithms. The new methods include the Greedy strategy, the ACO meta-heuristic, Stochastic Hill Climbing and our proposed, hybrid sk-ANT heuristic. Comprehensive experimental data revealed that our proposed method was the most effective and most stable of the methods, followed by the sk-strings heuristic. From this work we may conclude that the problem of structural inference is both important and tractable. For current applications we recommend use of the sk-ANT technique. The use of MML as a quality measure is also recommended, due to its generality and objectivity.

References

H Ahonen. *Generating Grammars for Structured Documents Using Grammatical Inference Methods.* Report A-1996-4, Department of Computer Science, University of Finland, 1996.

A W Biermann and J A Feldman. *On the synthesis of finite-state machines from samples of their behaviour.* IEEE Transactions on Computers, 21:591–597, 1972.

R C Carrasco and J Oncina (editors). *Grammatical Inference and Applications.* Proceedings of the Second International Colloquium on Grammatical Inference (ICGI-94), Lecture Notes in Artificial Intelligence 862, Springer-Verlag 1994.

R C Carrasco and J Oncina. *Learning Stochastic Regular Grammars by Means of a State Merging Method.* In R C Carrasco and J Oncina (editors) (Carrasco & Oncina 1994a).

J Chen. *Grammar Generation and Query Processing for Text Databases.* Research proposal, University of Waterloo, January 1991.

M Dorigo, V Maniezzo and A Coloni. *Positive Feedback as a Search Strategy.* Technical Report 91–016, Dipartmento di Elettronica, Politecnico di Milano, Italy, 1991.

P Fankhauser and Y Xu. *Markitup! An Incremental Approach to Document Structure Recognition.* Electronic Publishing - Origination, Dissemination and Design, 6(4):447–456, 1994.

M Garofalakis, A Gionis, R Rastogi, S Seshadri and K Shim. *XTRACT: A System for Extracting Document Type Descriptors from XML Documents.* In Proceedings of SIGMOD 2000, pages 165–176, Dallas, TX, 2000.

M P Georgeff and C S Wallace. *A General Selection Criterion for Inductive Inference.* In T O'Shea (editor), ECAI-84: Advances in Artificial Intelligence, pages 473–481. Dordretch: Elsevier, 1984.

L Pitt. *Inductive Inference, DFA's and Computational Complexity.* In J Siekmann (editor), Proceedings of the International Workshop AII '89, Lecture Notes in Artificial Intelligence 397, pages 18–44, Springer-Verlag 1989.

A V Raman and J D Patrick. *The sk-strings method for inferring PFSA.* In Proceedings of the 14th International Conference on Machine Learning, ICML'97, 1997.

A V Raman. *An Information Theoretic Approach to Language Relatedness.* PhD Thesis, Massey University, 1997.

Y Sakakibara. *Recent Advances in Grammatical Inference.* Theoretical Computer Science Volume 185, Number 1, October 1997.

J Sankey and R K Wong. *Structural Inference for Semistructured Data.* In Proceedings of the ACM International Conference on Information and Knowledge Management, CIKM'01, 2001.

K Shafer. *Creating DTDs via the GB-engine and Fred.* Available at **http://www.oclc.org/fred/**, 1995.

M D Young-Lai. *Application of a Stochastic Grammatical Inference Method to Text Structure.* Master's thesis, Computer Science Department, University of Waterloo, 1996.