JMLR: Workshop and Conference Proceedings 34:207–217, 2014 Proceedings of the 12th ICGI

Induction of Directed Acyclic Word Graph in a Bioinformatics Task

Wojciech Wieczorek WOJCIECH.WIECZOREK@US.EDU.PL
Institute of Computer Science, University of Silesia, Poland

Olgierd Unold OLGIERD.UNOLD@PWR.EDU.PL
Institute of Computer Engineering, Control and Robotics, Wroclaw University of Technology, Poland

Editors: Alexander Clark, Makoto Kanazawa and Ryo Yoshinaka

Abstract

In this paper a new algorithm for the induction of a Directed Acyclic Word Graph (DAWG) is proposed. A DAWG can serve as a very efficient data structure for lexicon representation and fast string matching, and have a variety of applications. Similar structures are being investigated in the theory of formal languages and grammatical inference, namely deterministic and nondeterministic finite automata (DFA and NFA, respectively). Since a DAWG is acyclic the proposed method is suited for problems where the target language does not necessarily have to be infinite. The experiments have been performed for a dataset from the domain of bioinformatics, and our results are compared with those obtained using the current state-of-the-art methods in heuristic DFA induction.

Keywords: Automata Induction, Grammatical Inference, Amyloid Proteins

1. Introduction

The automata play a central role in grammatical inference. The earliest theoretical results and practical methods in the field, concerned finite-state machines. Even now, when more and more work is being devoted to context-free grammars and other string-rewriting systems, a great deal of research is being conducted on automata induction methods. As this study will be especially focused on obtaining a directed acyclic word graph (which is a special case of an automaton) from finite positive and negative data, the various models of incremental learning and their decidability questions have not been mentioned. The book by de la Higuera (2010) can be of major help on such theoretical aspects of grammatical inference.

As far as algorithms for data-driven inductive inference of finite-state automata are concerned, the following classifications can be applied: the algorithms that output deterministic or non-deterministic automata, the class of languages they identify in the limit, and the exact methods (the inference of minimum size DFAs) or heuristic methods (the inference of automata which are just consistent with an input). The latter taxonomy is the one on which we base this work, as we are interested in the analysis of biological sequences, and only non-exact methods are suitable algorithms to process such large data sets. For instances of exact methods, the reader is referred to Biermann and Feldman (1972); Grinchtein et al. (2006); Heule and Verwer (2010). Heuristic methods which aim at selecting a small but not necessarily minimum size solution had marvelous success in DFA competitions (Lang

et al., 1998; Lucas and Reynolds, 2005). The most successful approaches are based on the idea of creating clusters of states to merge them in order to come up with a solution that is always consistent with the learning data. These methods, commonly known as state merging algorithms, start with the prefix tree acceptor for the training set and fold it up into a compact hypothesis by merging compatible pairs of states (Lang et al., 1998; Oncina and García, 1992; Trakhtenbrot and Barzdin, 1973).

With respect to the problem of DAWG construction one should draw attention to a number of material facts. There are also two types of DAWGs. A directed graph is called deterministic when no transitions exist that have the same labels and leave the same state. This property results in a very efficient search function. Graphs that do not have this property are called nondeterministic. The latter are generally smaller than the former but they are a little slower to search. Throughout the paper, we assume a DAWG to be nondeterministic. It is well known that NFA or a regular expression minimization (which are of a nondeterministic character, too) is computationally hard: it is PSPACE-complete (Meyer and Stockmeyer, 1972). Jiang and Ravikumar (1993) showed, moreover, that the minimization problem for NFAs or regular expressions remains PSPACE-complete, even when specifying the regular language by a DFA. Thus the problem of constructing a k-vertex DAWG that matches a set of input words is probably of exponential complexity. Some work has been done on the problem but for the set of examples only (i.e., without the presence of counter-examples and without the aim of generalization), see the algorithms devised by Amilhastre et al. (1999) and Sgarbas et al. (2001).

In the present algorithm a DAWG is achieved based on a learning sample containing the examples and counter-examples[1]. It is a two-phase procedure. In the first phase an initial graph is built in a way that resembles the construction of the minimal DFA, but nondeterminism is also allowed. In the second phase the graph is extended so as to increase the number of accepted words.

We have implemented our induction algorithm of a DAWG and started applying it to a real bioinformatics task, i.e. classification of amyloidogenic hexapeptides. Amyloids are proteins capable of forming fibrils instead of the functional structure of a protein (Jaroniec et al., 2004), and are responsible for a group of diseases called amyloidosis, such as Alzheimers, Huntingtons disease, and type II diabetes (Uversky and Fink, 2004). Furthermore, it is believed that short segments of proteins, like hexapeptides consisting of 6-residue fragments, can be responsible for amyloidogenic properties (Thompson et al., 2006). Since it is not possible to experimentally test all such sequences, several computational tools for predicting amyloid chains have emerged, inter alia, based on physico-chemical properties (Hamodrakas, 2011) or using machine learning approach (Stanislawski et al., 2013; Unold, 2012a,b). It is worth noting that proteins, but not amyloids, were recognized also by learning automata in (Coste and Kerbellec, 2006).

To test the performance of our DAWG approach, the following five programs have been used in experiments: an implementation of the Trakhtenbrot-Barzdin state merging algorithm, as described in Lang (1992); a MATLAB[2] implementation of the RPNI (Regular Positive and Negative Inference) algorithm (Akram et al., 2010); the implementations of two various versions of Rodney Price's Abbadingo winning idea of evidence-driven state

1. These examples and counter-examples are also called positive and negative words.
2. MATLAB is a registered trademark of The MathWorks, Inc.

merging (Lang et al., 1998); a program based on the Rlb state merging algorithm (Lang, 1997). The comparison to other classical automata learning algorithms such as ECGI (Rulot et al., 1989), DeLeTe (Denis et al., 2000), k-RI (Angluin, 1982), or k-TSSI (Garcia et al., 1990) needs further work. Also comparing with modified versions of the state merging algorithms, to ensure that returned automata are acyclic (by forbidding merges resulting in loops), deserves additional exploration.

The present paper is organized as follows. Section 2 gives the necessary definitions originating from graph theory and formal languages. Section 3 presents the proposed procedure of the construction of a DAWG. Section 4 discusses the preliminary experimental results of the new approach as well as evaluation of the methodology. Conclusions and research perspectives are contained in Section 5.

2. Definitions

A *directed graph*, or *digraph*, is a pair $G = (V, A)$, where V is a finite, non-empty set of *vertices* and A has as elements ordered pairs of different vertices called *arcs*; that is, $A \subseteq V \times V$. In a digraph $G = (V, A)$ the *in-degree* of a vertex v is the number of arcs of the form u, v that are in A. Similarly, the *out-degree* of v is the number of arcs of A that have the form v, u. A *walk* $w = (v_1, v_2, \ldots, v_k)$ of G is a sequence of vertices in V such that $(v_j, v_{j+1}) \in A$ for $j = 1, \ldots, k-1$. Furthermore, if $k > 1$ and $v_k = v_1$, then w is said to be *closed*. A *path* in G is a walk without repetitions. A *cycle* is a closed path.

An *alphabet* is a finite, non-empty set of symbols. We use the symbol Σ for an alphabet. A *word* (or sometimes *string*) is a finite sequence of symbols chosen from an alphabet. For a word w, we denote by $|w|$ the length of w. The *empty word* λ is the word with no symbols. Let x and y be words. Then xy denotes the *concatenation* of x and y, that is, a word formed by making a copy of x and following it by a copy of y. We denote as usual by Σ^* the set of words over Σ and by Σ^+ the set $\Sigma^* - \{\lambda\}$. A word w is called a *prefix* of a word x if there is a word u such that $x = wu$. It is a *proper* prefix if $u \neq \lambda$. The set of all prefixes that can be obtained from the set of words X will be denoted by $\mathcal{P}(X)$. Let X be a subset of Σ^*. For $w \in \Sigma^*$, we define the *left quotients* $w^{-1}X = \{u \in \Sigma^* \mid wu \in X\}$.

A DAWG $G = (s, t, V, A, \Sigma, \ell)$ is a digraph (V, A) with no cycles together with an alphabet Σ and with a label $\ell(u, v) \in 2^\Sigma - \{\emptyset\}$ for each $(u, v) \in A$, in which there is exactly one vertex with in-degree 0—the *source* s, and exactly one vertex with out-degree 0—the *terminal* t. It can be proved that in any DAWG, every vertex is reachable from the source. Furthermore, from every vertex the terminal is reachable. So there are no useless vertices. We will say that a word $w \in \Sigma^+$ is *stored* by G if there is a labeled path from the source to the terminal such that this path spells out the word w. Let $\mathcal{L}(G)$ be the set of all words that are spelled out by paths from s to t. We can formally define it by the *transition function* $\delta\colon V \times \Sigma^+ \to 2^V$ which is given inductively by

1. $\delta(v, a) = \{u \in V \mid (v, u) \in A \wedge a \in \ell(v, u)\}$, for $a \in \Sigma$,
2. $\delta(v, wa) = \bigcup_{i=1}^{n} \delta(u_i, a)$, where $\{u_1, u_2, \ldots, u_n\} = \delta(v, w)$, for $w \in \Sigma^+$ and $a \in \Sigma$.

Therefore, we have $\mathcal{L}(G) = \{w \in \Sigma^+ \mid t \in \delta(s, w)\}$. Using the terminology of automata, a DAWG could be referred to as a non-deterministic finite automaton that has exactly one accepting state and recognizes a finite language.

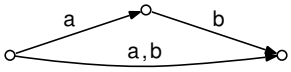

Figure 1: The initial DAWG built based on the set $\{a, b, ab\}$

Let $G = (s, t, V, A, \Sigma, \ell)$ be a directed acyclic word graph. To measure the *potency*, p, of $(u, v) \in A$ we make the following definition: $p(u, v)$ is the number of paths from v to t or 1 if $v = t$. Naturally, for all $v \in V - \{s, t\}$ we have $p(u, v) = \sum_{(v,w)\in A} p(v, w)$. Thus, by means of memoization, all potencies can be determined in time $O(|A|^2)$.

3. Efficient Construction of a DAWG from Informed Samples

A *sample* is a finite set of data. The sample can be *informed* in which case it is a pair (S_+, S_-) of finite sets of strings. S_+ contains the positive examples and S_- contains the negative examples (also called the counter-examples). Here and subsequently, we assume a sample to be non-conflicting (i.e., $S_+ \cap S_- = \emptyset$) and not containing the empty word.

The present algorithm is two-phased. In the first phase, based on the examples, an initial DAWG, G_{S_+}, is constructed. Its aim is to discover the structure of an unknown finite language. In the second phase the DAWG is extended step by step by the addition of new labels. The possibility of getting a DAWG that stores an expected language is the purpose of this extension procedure. So as to avoid the acceptance of a counter-example, the second phase is controlled by means of S_-.

An initial DAWG is constructed based on the following definitions. Its set of vertices is

$$V = \{w^{-1}S_+ \mid w \in \mathcal{P}(S_+)\}.$$

Its source is $s = S_+$, its terminal is $t = \{\lambda\}$, and its arcs and labels are defined for $v \in V$ and $a \in \Sigma$ by $a \in \ell(v, a^{-1}v) \Leftrightarrow a^{-1}v \neq \emptyset$ and $a \in \ell(v, t) \Leftrightarrow \lambda \in a^{-1}v$. From these definitions we can conclude that $\mathcal{L}(G_{S_+}) = S_+$. For example, the initial DAWG $G_{\{a,b,ab\}}$ is shown in Figure 1. Needless to say, in a computer implementation after building a digraph it is more convenient to represent its vertices as integers rather than sets of words.

We are now in a position to describe Procedure Extend, which, in an iterative process, improves the current word digraph by putting some additional labels onto the existing arcs. This auxiliary deposal leads in turn to an increase in the number of stored words and this is why we can get a DAWG that represents a more accurate or exact (sought) language. Please notice that the order of putting new labels alters the results, hence a greedy heuristic is used in order to obtain the most words consistent with a sample. The algorithm starts with the arcs of higher potency, then those of low potency are processed. The language membership problem (test if $t \in \delta(s, w)$) may be efficiently resolved by a memoized function derived from the definition of δ in time $O(|w| \cdot |V|^2)$.

Let n be the sum of the lengths of all examples, m be the sum of the lengths of all counter-examples, and k be the size of an alphabet. Since both $|V|$ and $|A|$ in G_{S_+} are $O(n)$, and the computation cost of determining V and A is in $O(n^2)$, the running time of the present method is $O(kmn^3)$. It is also noteworthy that an achieved DAWG stores only

Procedure Extend(G_{S_+}, S_-)

sort A according to decreasing potencies
for $(u, v) \in A$ **do**
  for $a \in \Sigma$ **do**
    if $a \notin \ell(u, v)$ **then**
      add a to $\ell(u, v)$
      if *there is a word* $w \in S_-$ *such that* $t \in \delta(s, w)$ **then**
        remove a from $\ell(u, v)$
      end
    end
  end
end

words of the same length as those observed in the set of examples. Therefore, it is pointless to have a negative example of length i, when none of the positive examples is of length i.

4. Experimental Results

In order to test our approach we considered the recently published Hexpepset dataset, containing a binary classification of 2452 hexapeptides made using the Pafig method (Tian et al., 2009). The original dataset contains 1226 positive samples and 1226 negative samples, however, according to Kotulska and Unold (2013), the Pafig method produces falsely correct results of classification. In Kotulska and Unold (2013) all 2452 hexapeptides of the Hexpepset dataset were applied to three state-of-the-art methods: FoldAmyloid (Garbuzynskiy et al., 2010) with five sets of different parameters, Waltz (Maurer-Stroh et al., 2010) with six sets of different parameters, and AmylPred (Hamodrakas et al., 2007). In total, all of these 12 methods returned 1676 instances unanimously classified (1648 negative samples and 28 positive samples)[3]. This set of unanimous classified hexapeptides was divided randomly into two equal training and test sets, each of them containing 14 positive examples and 824 counter-examples (denoted by the Limited training-test set).

To extend the dataset, we can use majority voting to all instances rejected by the above approach. Each hexapeptide is classified as a positive/negative example if most of 12 methods (i.e. $11, 10, \ldots, 7$) classifies it as a positive/negative, respectively. All examples which received an equal number of votes (of 6) are not considered here. As a result, we get 733 extra instances (611 negative and 122 positive samples). In this case the training set comprises all unanimous classified hexapeptides (1648 negative examples and 28 positive), and the testing set consists of all extra added 6-residue fragments (both sets are denoted by the Extended training-test set).

The DAWG approach was tested among five heuristic state-merging DFA induction algorithms: the Trakhtenbrot-Barzdin state merging algorithm (denoted Traxbar) (Lang, 1992), the RPNI algorithm (denoted RPNI) (Akram et al., 2010), two versions of evidence-

3. http://www.biomedcentral.com/1471-2105/14/351/additional, last accessed May 5, 2014.

driven state merging (denoted by Reference and Blue-fringe) (Lang et al., 1998), and the Rlb algorithm (denoted Rlb) (Lang, 1997)[4].

The classification results were evaluated based on typical measures: Recall, Precision, Balanced Accuracy, and area under the Receiver Operating Characteristic (ROC) curve (AUC). Recall and Precision are defined as follows:

Recall (true positive rate, Sensitivity) = tp/(tp+fn),
Specificity (true negative rate) = tn/(tn+fp),
Precision = tp/(tp+fp)

where tp, fp, fn and tn represent the numbers of true positives (correctly recognized amyloids), false positives (non-amyloids recognized as amyloids), false negatives (amyloids recognized as non-amyloids) and true negatives (correctly recognized non-amyloids), respectively.

Recall (completeness) in the context of amyloid recognition is the ratio of amyloids retrieved to the total number of amyloids in the dataset, Precision (exactness) is the fraction of the amyloids that are relevant to the find. Since the class distribution is not uniform among the classes both in Limited and Extended training-test sets (the set of positive hexapeptides forms a minority class), and to avoid inflated performance, the Balanced Accuracy and AUC were applied. Both of these metrics handle well with imbalanced datasets. The Balanced Accuracy (B.Acc) is defined as the arithmetic mean of Sensitivity and Specificity. AUC measures the area under the ROC curve (Fawcett, 2006), which represents the relationship between the true positive rate and the false positive rate of a bunch of classifiers resulting from different output thresholds[5]. However, in the case of binary classifiers, which we are faced with, the AUC may be computed easily using the following formula (Castro and Braga, 2012):

$$\mathrm{AUC}(f) = \frac{1}{|S_+| \times |S_-|} \left(\sum_{p \in S_+} \sum_{n \in S_-} G(f(p) - f(n)) \right),$$

where $G(t)$ is defined as

$$G(t) = \begin{cases} 0 & : \quad t < 0, \\ 0.5 & : \quad t = 0, \\ 1 & : \quad t > 0, \end{cases}$$

and assuming that $f(x) \in \{0, 1\}$ is dependent on mapping an instance x to negative (0) or positive (1) class. The AUC has an important statistical property: it is equivalent to the probability that a classifier f will rank a randomly chosen positive instance higher than a randomly chosen negative instance.

Table 1 summarizes the performances of the six compared methods. It transpires that the DAWG approach gained the highest Recall, Balanced Accuracy and AUC among tested heuristics, both for Limited and Extended training-test sets. It is worth noting, that in

4. We used implementations from the webpage http://code.google.com/p/gitoolbox/ for the RPNI and from http://abbadingo.cs.nuim.ie/dfa-algorithms.tar.gz for the remaining algorithms.
5. It is possible to choose different thresholds, if a classifier yields a numeric value (for instance probability or score) that represents the degree to which an object is a member of a class.

a (bio)medical context, Recall is regarded as primary score, as the goal of classification is to identify all real positive cases (i.e. in our case amyloids). What is more, both AUC metric and B.Acc sort the compared methods in the same order, almost irrespective of the training-test set applied (the only difference is the highest position of the Reference state merging among methods over the Extended set). This demonstrates the high correlation of the two metrics, as well as the similar characteristics of the two training-test sets. Note that the extended training set includes more examples, but the extended testing set is based on less reliable data than the limited set.

Due to the method of constructing Limited training-test set, the presented results over this dataset can be considered as a comparision with dedicated bioinformatics tools: FoldAmyloid (Garbuzynskiy et al., 2010), Waltz (Maurer-Stroh et al., 2010) and AmylPred (Hamodrakas et al., 2007). All these mentioned approaches classified hexapeptides faultlessly, and as such gained the highest scores of Precision, Recall, Balanced Accuracy, and AUC metrics.

Table 1: Comparison of DAWG with other heuristic DFA induction methods over limited and extended training-test sets. P=Precision, R=Recall, B.Acc=Balanced Accuracy, AUC=area under the ROC curve. The table is arranged in order of decreasing Balanced Accuracy regarding Limited training-test set.

	Limited training-test set				Extended training-test set			
Method	P	R	B.Acc	AUC	P	R	B.Acc	AUC
DAWG	0.2034	**0.8571**	**0.9273**	**0.9001**	0.4038	**0.5164**	**0.7071**	**0.6821**
Rlb	0.6875	0.7857	0.8910	0.8898	0.6232	0.3525	0.6167	0.6549
Traxbar	0.5238	0.7857	0.8910	0.8868	0.4706	0.3279	0.6007	0.6271
Reference	0.4545	0.7142	0.8547	0.8499	0.4757	0.4016	0.6429	0.6566
Blue-fringe	1.0	0.5	0.7458	0.75	0.6486	0.1967	0.5279	0.5877
RPNI	0.6667	0.4285	0.7095	0.7125	0.7917	0.1557	0.5052	0.5738

It would be interesting to test the performance of DFA induction methods, including DAWG approach, on only experimentally asserted amyloid sequences. Table 2 presents the results obtained by the compared methods on amyloidogenic dataset, composed by 116 hexapeptides known to induce amyloidosis and by 161 hexapeptides that do not induce amyloidosis (Maurer-Stroh et al., 2010). The set of experimentally verified hexapeptides was divided randomly into two equal training and test sets, each of them containing 58 positive examples and 80/81 (training/testing) counter-examples. It is not hard to notice, that experimental set is much smaller than computational made datasets (one order of magnitude) and better balanced. Nevertheless, the DAWG approach achieved best Recall and Balanced Accuracy metrics, rightly predicting 94.8% of the actual amyloid cases, and as such can be used with a high reliability to recognize amyloid proteins.

5. Conclusions and Future Work

We have proposed an efficient method for inferring a directed acyclic word graph from finite positive and negative data. The idea behind the method is to build a nondeterministic word

Table 2: Comparison of DAWG with other heuristic DFA induction methods over experimental verified set. P=Precision, R=Recall, B.Acc=Balanced Accuracy, AUC=area under the ROC curve. The table is arranged in order of decreasing Balanced Accuracy.

Method	Experimental set			
	P	R	B.Acc	AUC
DAWG	0.4264	**0.9483**	**0.8241**	0.5173
Traxbar	0.5658	0.7414	0.7515	0.6669
RPNI	0.6667	0.4285	0.7095	0.7125
Rlb	0.5	0.6207	0.6462	0.5881
Blue-fringe	0.4019	0.7069	0.6237	0.4769
Reference	0.3402	0.5689	0.4869	0.3894

graph that stores exactly the example set, and then to extend it by putting new transition labels in a manner that maximizes the set of stored (accepted) words, while assuring that none of the counter-examples is accepted.

In this work, we have used data derived from the recently published Hexpepset dataset and experimentally verified dataset for the inference of a DAWG among the other five heuristic state-merging DFA induction algorithms. The DAWG approach outperformed significantly all compared heuristic methods in terms of Recall and Balanced Accuracy for all tested sets (and in term of AUC for computational made sets). As a by-product, we also propose a more justified—by using unanimous classification by three state-of-the-art methods—amyloid hexapeptide dataset (called Limited training-test set) than the reference Hexpepset.

The work we are carrying out at present is the application of the proposed method not only to recognize amyloid proteins, but also to generate them. Observing that having the statistical distribution of aminoacids in hexapeptides (Kotulska and Unold, 2013), we are able to produce by means of an induced directed acyclic word graph a set of new likely fibril-forming segments in the protein sequences. The amyloidogenicity of generated sequences can be later tested, for example by a simplified 3D profile method (Stanislawski et al., 2013).

Acknowledgments

This research was supported in part by PL-Grid Infrastructure, and the grants DEC-2011/03/B/ST6/01588 and N N519 643540 from National Science Center of Poland.

References

H. I. Akram, C. de la Higuera, H. Xiao, and C. Eckert. Grammatical inference algorithms in MATLAB. In *Proceedings of the 10th International Colloquium Conference on Grammatical Inference: Theoretical Results and Applications*, pages 262–266. Springer-Verlag, 2010.

J. Amilhastre, P. Janssen, and M. Vilarem. FA minimisation heuristics for a class of finite languages. In *WIA*, volume 2214 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 1999.

D. Angluin. Inference of reversible languages. *Journal of the ACM*, 29(3):741–765, 1982.

A. W. Biermann and J. A. Feldman. On the synthesis of finite-state machines from samples of their behavior. *IEEE Trans. Comput.*, 21(6):592–597, 1972.

C. L. Castro and A. P. Braga. Improving ANNs performance on unbalanced data with an AUC-based learning algorithm. In *ICANN (2)*, volume 7553 of *Lecture Notes in Computer Science*, pages 314–321. Springer, 2012.

F. Coste and G. Kerbellec. Learning automata on protein sequences. In *Proceedings of the 7th Journées Ouvertes Biologie Informatique Mathématiques 2006 (JOBIM 2006)*, pages 199–210, 2006.

C. de la Higuera. *Grammatical Inference: Learning Automata and Grammars*. Cambridge University Press, 2010.

F. Denis, A. Lemay, and A.Terlutte. Learning regular languages using non deterministic finite automata. In Arlindo L. Oliveira, editor, *ICGI*, volume 1891 of *Lecture Notes in Computer Science*, pages 39–50. Springer, 2000.

T. Fawcett. An introduction to ROC analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

S. O. Garbuzynskiy, M. Y. Lobanov, and O. V. Galzitskaya. FoldAmyloid: a method of prediction of amyloidogenic regions from protein sequence. *Bioinformatics*, 26(3):326–332, 2010.

P. Garcia, E. Vidal, and J. Oncina. Learning locally testable languages in the strict sense. In *ALT*, pages 325–338, 1990.

O. Grinchtein, M. Leucker, and N. Piterman. Inferring network invariants automatically. In *Proceedings of the Third International Joint Conference on Automated Reasoning*, pages 483–497. Springer-Verlag, 2006.

S. J. Hamodrakas. Protein aggregation and amyloid fibril formation prediction software from primary sequence: towards controlling the formation of bacterial inclusion bodies. *Febs Journal*, 278(14):2428–2435, 2011.

S. J. Hamodrakas, C. Liappa, and V. A. Iconomidou. Consensus prediction of amyloidogenic determinants in amyloid fibril-forming proteins. *International journal of biological macromolecules*, 41(3):295–300, 2007.

M. J. H. Heule and S. Verwer. Exact DFA identification using SAT solvers. In *Proceedings of the 10th International Colloquium Conference on Grammatical Inference: Theoretical Results and Applications*, pages 66–79. Springer-Verlag, 2010.

C. P. Jaroniec, C. E. MacPhee, V. S. Bajaj, M. T. McMahon, C. M. Dobson, and R. G. Griffin. High-resolution molecular structure of a peptide in an amyloid fibril determined by magic angle spinning NMR spectroscopy. *Proceedings of the National Academy of Sciences of the United States of America*, 101(3):711–716, 2004.

T. Jiang and B. Ravikumar. Minimal NFA problems are hard. *SIAM Journal on Computing*, 22:1117–1141, 1993.

M. Kotulska and O. Unold. On the amyloid datasets used for training PAFIG-how (not) to extend the experimental dataset of hexapeptides. *BMC Bioinformatics*, 14(1):351, 2013.

K. J. Lang. Random DFA's can be approximately learned from sparse uniform examples. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 45–52. ACM, 1992.

K. J. Lang. Merge Order count. Technical report, NECI, 1997.

K. J. Lang, B. A. Pearlmutter, and R. A. Price. Results of the Abbadingo One DFA learning competition and a new evidence-driven state merging algorithm. In *Proceedings of the 4th International Colloquium on Grammatical Inference*, pages 1–12. Springer-Verlag, 1998.

S. M. Lucas and T. J. Reynolds. Learning deterministic finite automata with a smart state labelling evolutionary algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1063–1074, 2005.

S. Maurer-Stroh, M. Debulpaep, N. Kuemmerer, M. Lopez de la Paz, I. C. Martins, J. Reumers, K. L. Morris, A. Copland, L. Serpell, L. Serrano, et al. Exploring the sequence determinants of amyloid structure using position-specific scoring matrices. *Nature methods*, 7(3):237–242, 2010.

A. R. Meyer and L. J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *Proceedings of the 13th Annual Symposium on Switching and Automata Theory*, pages 125–129, 1972.

J. Oncina and P. García. Identifying regular languages in polynomial time. In H. Bunke, editor, *Advances in Structural and Syntactic Pattern Recognition*, volume 5 of *Machine Perception and Artificial Intelligence*, pages 99–108. World Scientific, 1992.

H. Rulot, N. Prieto, and E. Vidal. Learning accurate finite-state structural models of words through the ecgi algorithm. In *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, pages 643–646 vol.1, 1989.

K. N. Sgarbas, N. Fakotakis, and G. K. Kokkinakis. Incremental construction of compact acyclic NFAs. In *ACL*, pages 474–481. Morgan Kaufmann Publishers, 2001.

J. Stanislawski, M. Kotulska, and O. Unold. Machine learning methods can replace 3D profile method in classification of amyloidogenic hexapeptides. *BMC Bioinformatics*, 14 (1):21, 2013.

M. J. Thompson, S. A. Sievers, J. Karanicolas, M. I. Ivanova, D. Baker, and D. Eisenberg. The 3D profile method for identifying fibril-forming segments of proteins. *Proceedings of the National Academy of Sciences of the United States of America*, 103(11):4074–4078, 2006.

J. Tian, N. Wu, J. Guo, and Y. Fan. Prediction of amyloid fibril-forming segments based on a support vector machine. *BMC Bioinformatics*, 10(Suppl 1):S45, 2009.

B. Trakhtenbrot and Y. Barzdin. *Finite automata: behavior and synthesis*. North-Holland Publishing Company, 1973.

O. Unold. Fuzzy grammar-based prediction of amyloidogenic regions. *Journal of Machine Learning Research-Proceedings Track*, 21:210–219, 2012a.

O. Unold. How to support prediction of amyloidogenic regions-the use of a GA-based wrapper feature selections. In *IMMM 2012, The Second International Conference on Advances in Information Mining and Management*, pages 37–40, 2012b.

V. N. Uversky and A. L. Fink. Conformational constraints for amyloid fibrillation: the importance of being unfolded. *Biochimica et Biophysica Acta (BBA): Proteins and Proteomics*, 1698(2):131–153, 2004.