

Biological signal prediction using stochastic regular grammars

André Y. Kashiwabara and Alan M. Durham — Departamento de Ciência da Computação (IME)

UNIVERSIDADE DE SÃO PAULO

INTRODUCTION

A stochastic regular grammar (SRG), also called probabilistic finite automata (PFA), is a “syntactic object” that can generate the same probability distribution as hidden Markov model over a language [7]. One important characteristic of SRGs is the existence of algorithms that can infer the automata’s architecture from a training sample. We will show that the theoretical search space of deterministic PFA (DPFA) inference algorithms such as LAPFA [5] and ALERGIA [2] contain two models widely applied in gene prediction programs: weight array method (WAM) [9] and weight matrix model (WMM) [6, 1]. We also show that, with small modifications both WAM and WMM are in the effective search space of LAPFA. With this result we show that SRGs can substitute the use of the other two technologies, with the advantage that the selection of the appropriate model is performed automatically by the inference algorithm, based on the characteristics of the training sample, and not a priori by the researcher.

MATHEMATICAL MODELS

Probabilistic Finite Automata

A probabilistic finite automata (PFA) is a 6-tuple $A = (Q, \Sigma, \delta, I, F, P)$ [7] where:

- Q is the set of state
- Σ is the finite alphabet
- $\delta \subseteq Q \times \Sigma \times Q$ is the set of transitions.
- $I : Q \rightarrow [0, 1]$ is the initial-state probability function
- $P : \delta \rightarrow [0, 1]$ is the transition probability function
- $F : Q \rightarrow [0, 1]$ is the final-state probability function

The functions I , P , and F have the following properties:

$$\sum_{q \in Q} I(q) = 1$$

and for all $q_1 \in Q$, $\sum_{\sigma \in \Sigma, q_2 \in Q} P(q_1, \sigma, q_2) + F(q_1) = 1$

In particular, a PFA is deterministic, if:

- There is a q_0 , called initial state, such that $I(q_0) = 1$
- For all $q \in Q$, and for all $\sigma \in \Sigma$ $|\{q' : (q, \sigma, q') \in \delta\}| \leq 1$

Figure 1 contains an example of a DPFA and a non-deterministic PFA.

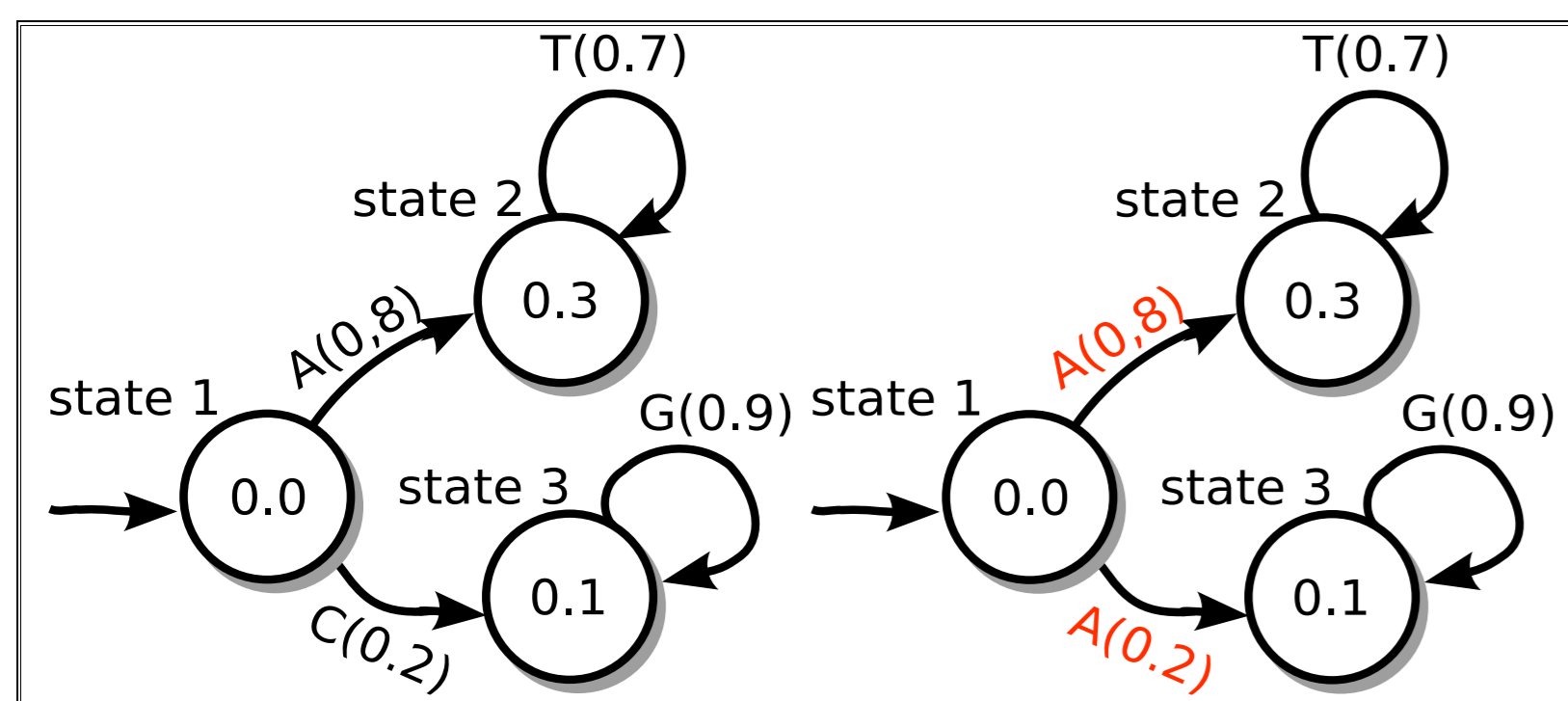


Figure 1. We have two probabilistic finite automata: a deterministic PFA at the left, and a non-deterministic PFA at the right. The number inside the state is the final probability. For each transition we have an emission symbol and a transition probability. The non-deterministic PFA has two transitions with the same symbol from the same state.

DPFA parser

The probability of the sequence $S = s_0 \dots s_{n-1}$ is given by:

$$P_{DPFA}(S) = \left\{ \prod_{i=1}^n P(q_{i-1}, s_{i-1}, q_i) \right\} F(q_n)$$

The sequence of states q_0, \dots, q_n is the only path over the DPFA given the sequence S .

Weight Matrix Model

Weight matrix model is a simple mathematical model that captures the positional frequencies of the biological signal. It assumes the independence between positions [1].

Let $p_i(s_i)$ be the probability of the symbol s_i at position i , the probability of a sequence $S = s_0 \dots s_{n-1}$ be generated by this model is given by:

$$P_{WMM}(S) = \prod_{i=0}^{n-1} p_i(s_i)$$

Weight Array Method

Weight array method is an inhomogeneous Markov chain. It captures the correlation between adjacent positions of the signal [1].

Let $p_i(s_i|s_{i-1})$ be the probability of the symbol s_i at position i given the symbol s_{i-1} at position $i-1$, the probability of a sequence $S = s_0 \dots s_{n-1}$ be generated by this model is given by:

$$P_{WAM}(S) = p_0(s_0) \prod_{i=1}^{n-1} p_i(s_i|s_{i-1})$$

PFA INFERENCE SEARCH SPACE

The search space of DPFA inference algorithm is a lattice [3]

DPFA inference algorithms work over a search space that can be viewed as a lattice where the elements are the automata. The canonical automata of this lattice is the prefix tree automata (Figure 2) that recognizes only the training set, and the universal automata has only one state with recursive transitions for every entry symbol, recognizing any string over the alphabet. The DPFA inference algorithms mentioned above modify the prefix tree automata interactively by merging different states. This creates a search space of automata that can be reached from the prefix tree automata by successive “state merging operations” [2].

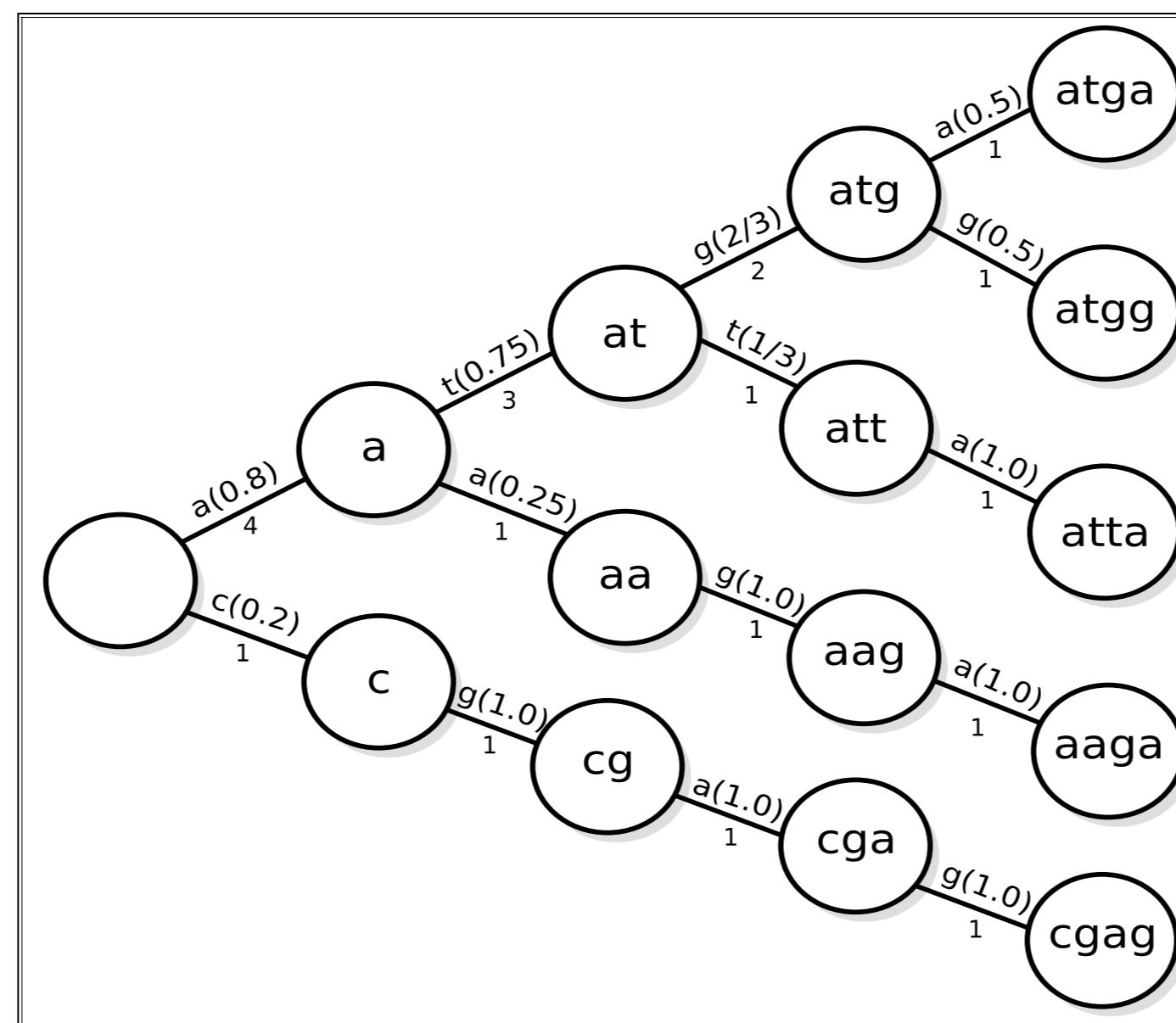


Figure 2. A probabilistic tree automata for the training set $\{atga, atgg, atta, aaga, cgag\}$. Only this training set can be recognized by this automata.

Weight matrix model is in the DPFA inference algorithm search spaces

We can fold together all states that are in the same level, the resulting probabilistic automaton will be equivalent to a WMM. Figure 3 shows an example of a DPFA that is equivalent to a WMM.

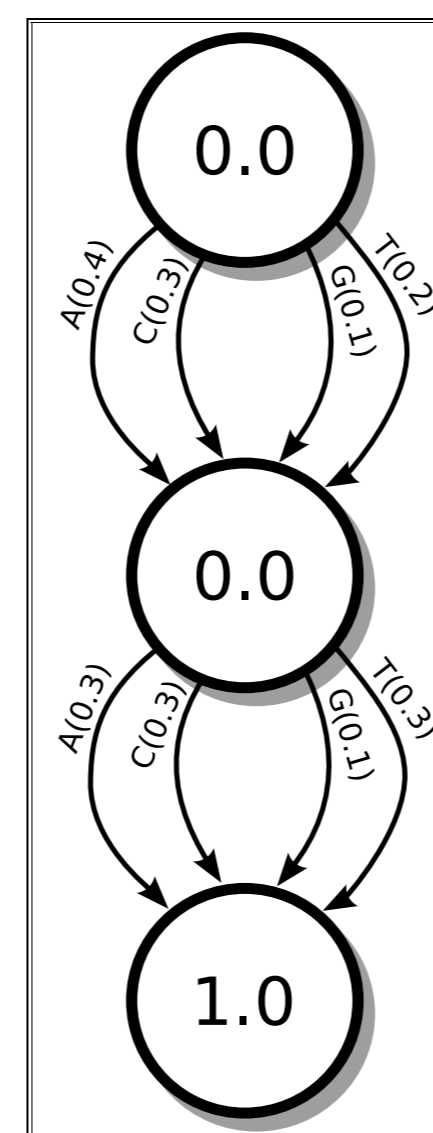


Figure 3. This is an example of an universal automata to the LAPFA search space. This automata generates the same probability distribution as a WMM with the following parameters: $p_0(A) = 0.4$, $p_0(C) = 0.3$, $p_0(G) = 0.1$, $p_0(T) = 0.1$, $p_1(A) = 0.3$, $p_1(C) = 0.3$, $p_1(G) = 0.1$, and $p_1(T) = 0.3$.

Weight array method is in the DPFA inference algorithm search spaces

Some automata in the lattice generate the same probability distribution over a language as WAM. This automata can be reached by merging together only the states, from the same level, that have incoming transitions with the same label. Figure 4 shows a PFA with the same probability distribution as a WAM with the following parameters: $p_0(0) = 0.5$, $p_0(1) = 0.5$, $p_1(0|0) = 0.25$, $p_1(1|0) = 0.75$, $p_1(0|1) = 0.75$, $p_1(1|1) = 0.25$, $p_2(0|0) = 0.7$, $p_2(1|0) = 0.3$, $p_2(0|1) = 0.3$, $p_2(1|1) = 0.7$, $p_3(0|0) = 0.8$, $p_3(1|0) = 0.2$, $p_3(0|1) = 0.2$, and $p_3(1|1) = 0.8$

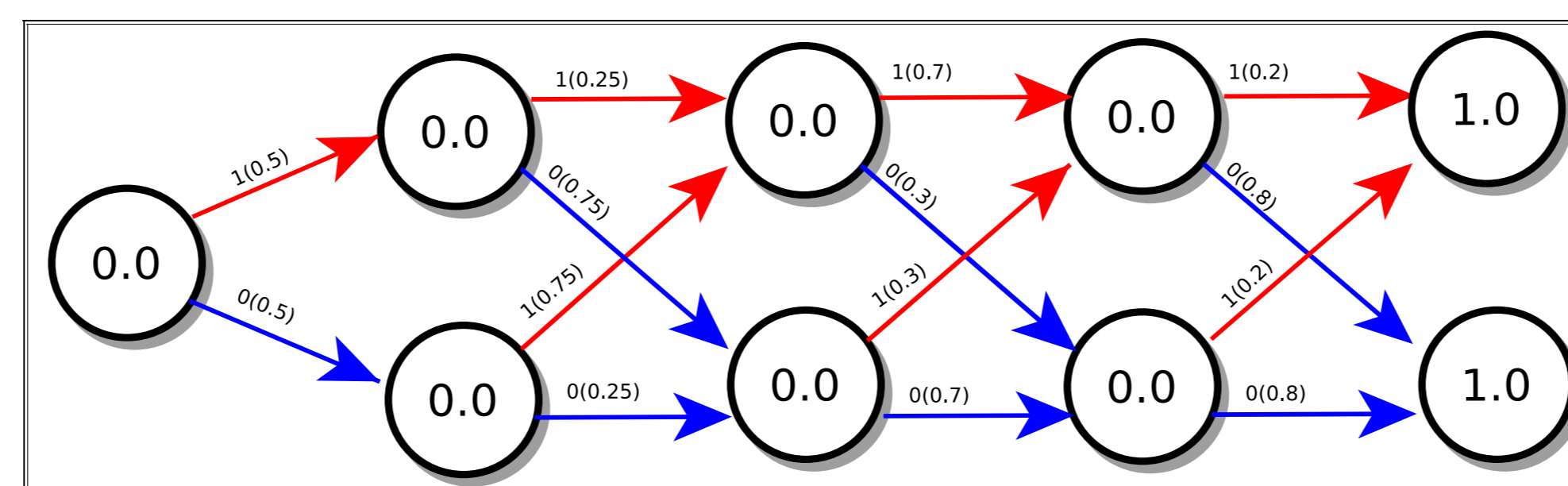


Figure 4. This example shows a PFA equivalent to a WAM that can be reached by the LAPFA algorithm. This PFA generates a probability distribution over the language Σ^4 , $\Sigma = \{0, 1\}$.

DPFA inference algorithm search space contains DPFA with different memory length in each position

The automata that are generated may be richer than a simple WAM. Some automata in the search space have variable memory length, and others can have complex correlations between states.

Figure 5 shows an automata with variable memory length.

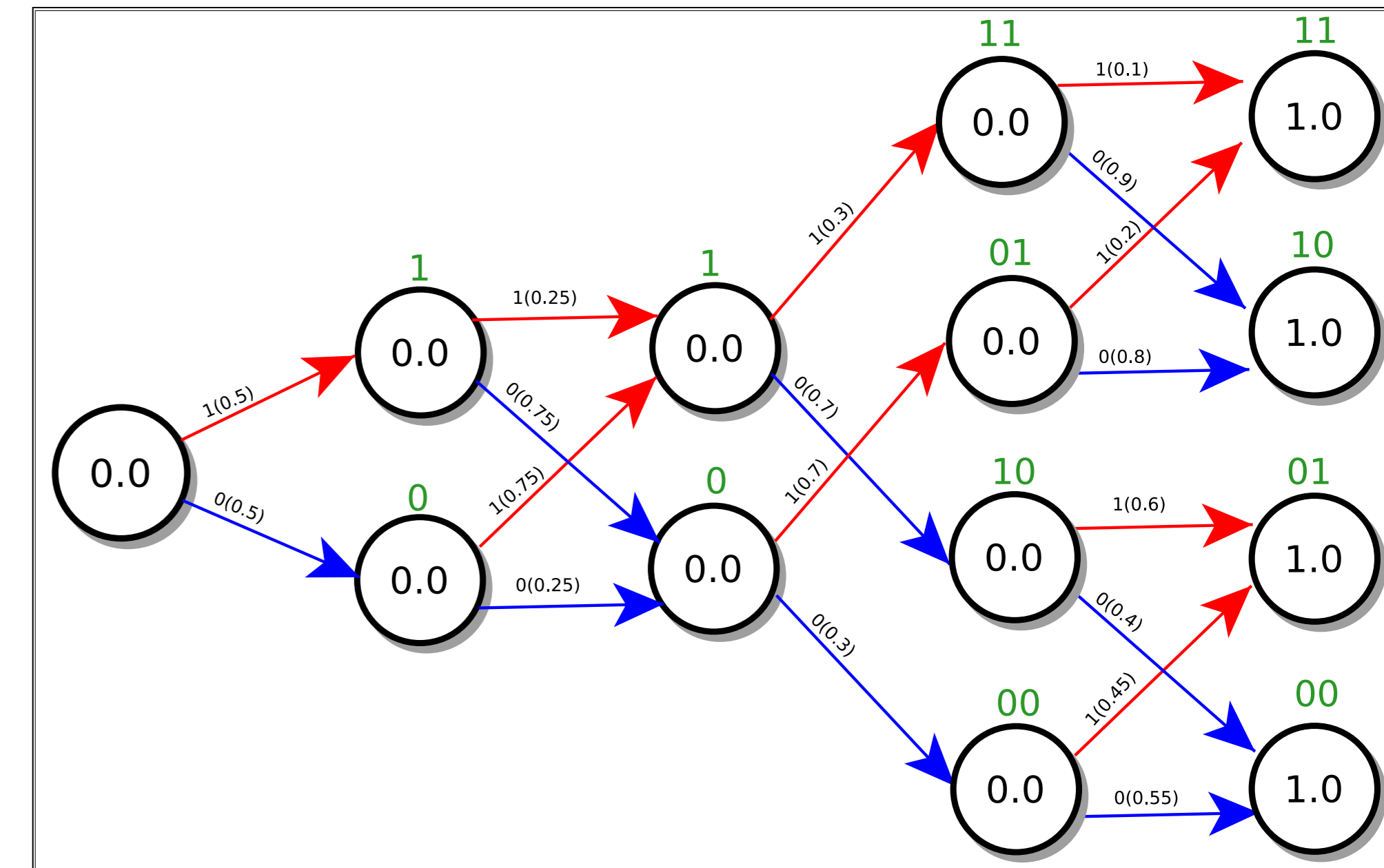


Figure 5. This PFA generates a probability distribution over the language Σ^4 ($\Sigma = \{0, 1\}$). The memory length is variable and dependent of the position: green numbers represent the memory of each state.

LAPFA search space is restricted by acyclic automata

LAPFA algorithm maintains a sequence of probabilistic acyclic finite automata, the first automata is the prefix tree automata, and the last automata is a hypothesis acyclic DPFA. When two states are found to be “similar”, the automata in this sequence is transformed into another automata by merging these two similar states. LAPFA will merge only states from the same level [5].

The universal automaton is the smallest automata that can be reached, and it has only n states where n is the size of the greatest sequence in the training set. In the biological signal prediction problem, all sequences have the same size n . Figure 3 is an example of universal probabilistic automaton from a LAPFA’s lattice.

A LAPFA modification

LAPFA algorithm ensure that the sequence of automata converges to a good approximation of the target automata [5], but this approximation is not necessary the one with the best accuracy at biological signal prediction.

In the problem of biological signal prediction by computational methods, the models that capture the dependencies between positions have better accuracy. In general, WAM is better than WMM when we have sufficient data [1]. The LAPFA modification we have implemented ensures that the sequence of DPFA contains automata with the same distribution of WAMs of different order. For each step of the algorithm, two states are merged together if they are similar and have incoming transitions with the same label. This iterative procedure stops when there are no similar states to merge, and at the end of the algorithm, it will join together all states from the same level that has a data size less than m_0 . This modification ensures a convergence to a WAM, but the effective search space is now smaller. If the training set is small, then this modification will merge all states from the same level, generating a model equivalent to a WMM.

CONCLUSION AND FUTURE WORK

In a previous work, we have shown that LAPFA algorithm can generate DPFA with similar performance than NNSPLICE [4] for splicing site prediction [8]. In this work, we have shown why it is true. In particular, LAPFA can generate automata that are similar to a WMM, or a WAM.

We have to improve this modification presented in this poster. The modification we have presented did not see the correlations between positions. Another possible modification may be the inclusion of a χ^2 test that will analyze these correlations.

ACKNOWLEDGEMENT

We would like to acknowledge CAPES for finance this research.

REFERENCES

- [1] C. Burge. Modeling dependencies in pre-mRNA splicing signals. *Computational Methods in Molecular Biology*, 32:129–164, 1998.
- [2] R. C. Carrasco and J. Oncina. Learning stochastic regular grammars by means of a state merging method. In *International Conference on Grammatical Inference*, pages 999–999. Springer-Verlag, September 1994.
- [3] P. Dupont, L. Miclet, and E. Vidal. What is the search space of the regular inference? In *ICGI: International Colloquium on Grammatical Inference and Applications*, 1994.
- [4] M. G. Reese and F. H. Eeckman. Improved splice site detection in Genie. *J Comp Biol*, 4:311–323, 1997.
- [5] D. Ron, Y. Singer, and N. Tishby. On the learnability and usage of acyclic probabilistic finite automata. *JCSS: Journal of Computer and System Sciences*, 56, 1998.
- [6] R. Staden. Computer methods to locate signals in nucleic acid sequences. *Nucleic Acids research*, 12:505–519, 1984.
- [7] E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, and R. C. Carrasco. Probabilistic finite-state machines—part I. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27, 2005.
- [8] D. da Cruz Vieira, A. Y. Kashiwabara, A. M. Lima, and A. M. Durham. Splice site prediction using stochastic regular grammars. *International Conference of Bioinformatics and Computational Biology*, 2004.
- [9] M. Q. Zhang and T. G. Marr. A weight array method for splicing signal analysis. *Computer Applied in Bioscience*, 9:499–509, 1993.