TROPICAL MODELING OF WEIGHTED TRANSDUCER ALGORITHMS ON GRAPHS

Emmanouil Theodosis and Petros Maragos

School of ECE, National Technical University of Athens, 15773 Athens, Greece
manostheodosis@mail.ntua.gr, maragos@cs.ntua.gr

ABSTRACT

Weighted Finite State Transducers (WFSTs) are versatile graphical automata that can model a great number of problems, ranging from automatic speech recognition to DNA sequencing. Traditional computer science algorithms are employed when working with these automata in order to optimize their size, but also the run time of decoding algorithms. However, these algorithms are not unified under a common framework that would allow for their treatment as a whole. Moreover, the inherent geometrical representation of WFSTs, coupled with the topology-preserving algorithms that operate on them make the structures ideal for tropical analysis. The benefits of such analysis have a twofold nature; first, matrix operations offer a connection to nonlinear vector space and spectral theory, and, second, tropical algebra offers a connection to tropical geometry. In this work we model some of the most frequently used algorithms in WFSTs by using tropical algebra; this provides a theoretical unification and allows us to also analyze aspects of their tropical geometry.

Index Terms— Weighted Finite State Transducers, tropical algebra, tropical geometry

1. INTRODUCTION

Weighted Finite State Transducers (WFSTs) are graphical automata which find application in many fields ranging from language and speech processing to computational biology. There exists a multitude of algorithms that operate on WFSTs [1]–[3]. The most prominent and most studied is the Viterbi algorithm [4]–[6], which stems from the field of telecommunications, and its variants. However, such algorithms are usually computationally expensive, which is undesirable for practical applications. Besides algorithms that simply utilize a WFST, there are more intrusive algorithms that alter its parametrization in an effort to optimize subsequent decoding, while maintaining the inherent structure, such as the weight pushing [1]–[3] and the epsilon removal [1]–[3] algorithms, stemming from computer science. Some of these algorithms aim to directly reduce the number of states and arcs in WFSTs, and thus immediately affecting the time requirements of the decoding. Others try to indirectly affect the execution speed, by reweighting the arcs between states so that pruning algorithms examine fewer paths.

These algorithms admit modeling through tropical algebra and tropical geometry [1]–[3], [7], [8], however no efforts have been made to thoroughly explore their tropical aspects beyond the expression of scalar arithmetic with operations from the tropical semiring. For detailed background on the tropical semiring and max-plus algebra we refer the reader to [9]–[16]. In this paper we model the algorithms using tropical algebra and related min-plus matrix operations, resulting in novel expressions in closed matrix form. We also explain aspects of the geometry of certain algorithms, namely the Viterbi pruning.

References [1]–[3] are some of the most influential of the field, studying the WFST structures and proposing the corresponding algorithms. Minimax algebra and its application to scheduling was introduced in [11]. Max-plus algebra and its applications to control theory and optimization have been studied in [12]–[14], [17]. Some recent related works from our group include a study of systems on weighted lattices as a unification of max-plus algebra and its generalizations [16]; modeling nonlinear perceptrons with max-plus algebra and tropical geometry [18]; modeling the Viterbi algorithm and its pruning variant in min-plus algebra [19]; and generalization of tropical geometry using weighted lattices [20].

In this paper we provide a theoretical unification of WFSTs algorithms by modeling them using tropical algebra, which also allows for their further analysis using tools from minimax matrix theory [11]. We first model the weight pushing algorithm, a non-intrusive algorithm that aims to speed up pruning by propagating the weights to earlier states of the WFST. Then we model the epsilon removal algorithm, which alters the structure of the WFST in order to remove unnecessary states and transitions, thus reducing its size and immediately affecting decoding. We present previous results regarding the modeling of the Viterbi algorithm and its pruning variant. Finally, we further explore the properties of certain metrics defined through the Viterbi pruning and elaborate on their motivation. Our modeling aspires to offer a connection with, and unification via, the nonlinear vector space theory of weighted lattices [16] and aspires to allow for spectral analysis of these algorithms. In addition, we provide links with tropical geometry, similar to the efforts in [18], [19].

In Section 2 we present elements of tropical algebra that will be useful in our analysis. Section 3 contains the modeling of the various algorithms in tropical algebra; the weight pushing, epsilon removal, and Viterbi algorithms. Finally, in Section 4 we revisit the geometry of the Viterbi pruning and we better explain the motivation for and the properties of metrics defined in previous work [19].

2. BACKGROUND

Tropical algebra is similar to linear algebra. Like linear algebra studies systems of linear equations and their properties, tropical algebra studies systems of nonlinear equations (min-plus equations) and their properties. The main pair of operations is the pair $(\min, +)$, and we will use $\wedge$ to denote the minimum. The vectors and matrices of tropical algebra exist on the extended real multidimensional space defined by $\mathbb{R}_{\min} = \mathbb{R} \cup \{+\infty\}$. In this paper, we follow the notation of [16] for the operations on weighted lattices. Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}_{\min}^{n \times m}$.

This research has been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project code:T1EDK-02890, acronym: e-Prevention).

Their min-plus product, denoted by $\boxplus$, is given by:

$$(\mathbf{A} \boxplus \mathbf{B})_{ij} = \bigwedge_{k=1}^{m} A_{ik} + B_{kj} \tag{1}$$

We will also make extensive use of two important matrices:

- the *weak transitive closure* $\Gamma(\cdot)$ of a matrix $\mathbf{A}$, defined as:

$$\Gamma(\mathbf{A}) = \mathbf{A} \wedge \mathbf{A}^2 \wedge \ldots \wedge \mathbf{A}^n \wedge \ldots \tag{2}$$

- the *strong transitive closure* $\Delta(\cdot)$ of a matrix $\mathbf{A}$, defined as:

$$\Delta(\mathbf{A}) = \mathbf{I} \wedge \mathbf{A} \wedge \mathbf{A}^2 \wedge \ldots \wedge \mathbf{A}^n \wedge \ldots \tag{3}$$

We can see that $\Delta(\mathbf{A}) = \mathbf{I} \wedge \Gamma(\mathbf{A})$. These two matrices provide solutions to tropical eigenvector problems:

- the weak transitive closure $\Gamma(\mathbf{A})$ provides solutions to the min-plus eigenvector-eigenvalue problem $\mathbf{A} \boxplus \mathbf{x} = \lambda + \mathbf{x}$.
- $\Delta(\mathbf{A})$ provides solutions to the generalized min-plus eigenvector-eigenvalue problem $\mathbf{A} \boxplus \mathbf{x} \geq \lambda + \mathbf{x}$.

Tropical geometry [21]–[23] aims to generalize the ideas of Euclidean geometry to the tropical setting. This proves useful in many cases because tropical curves are piecewise linear, which offers immediate bounds for the solution space of problems, but also offers ties to linear programming [24]–[26] and its algorithms. Similar to its Euclidean counterpart, a *tropical line* is given by

$$y = \alpha + x \wedge \beta = \min(\alpha + x, \beta) \tag{4}$$

We can also define tropical halfspaces and polytopes as:

Definition 1. *An **affine tropical halfspace**, parametrized by $\mathbf{a}, \mathbf{b} \in \mathbb{R}_{\min}^{n+1}$, is a subset of $\mathbb{R}_{\min}^{n}$ defined by:*

$$T(\mathbf{a}, \mathbf{b}) := \{\mathbf{x} : \min_i (a_i + x_i) \wedge a_{n+1} \geq \min_i (b_i + x_i) \wedge b_{n+1}\}$$

Definition 2. *A bounded intersection of a finite number of tropical halfspaces is will be called a **tropical polytope**.*

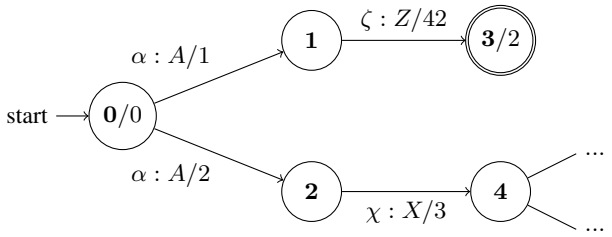
We will also use the epsilon closure from automata theory [27]:

Definition 3. *The **epsilon closure** of a state q is the set of all the states reachable by q with epsilon transitions.*

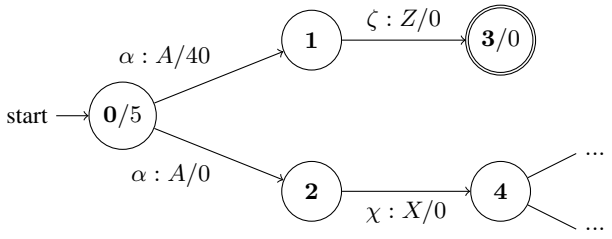
3. MODELING

3.1. Weight pushing

The weight pushing algorithm is an essential algorithm for practical application of the WFST framework. The algorithm aims to propagate the weights to earlier states of the structure, so that low-probability paths are recognized earlier in the decoding sequence, and thus have a higher chance of being pruned by pruning algorithms. An irrevocable requirement is that the underlying structure of the WFST must remain the same: the algorithm might alter the weights, but the set of accepted paths and their total weights must stand unaffected. An example highlighting the weight pushing operation appears in Fig. 1. An improbable path that has, at an early stage, a low cost will consume computational resources, where that could have been avoided by pushing the overall weight in earlier transitions. The algorithm can be divided into two parts; the calculation of each state's potential (the weight amount that can be propagated to earlier states), and the update of the WFST's weights. A



(a) WFST before the weight pushing operation.



(b) WFST after the weight pushing operation.

Fig. 1: WFST transducing a sequence of lowercase Greek letters to capital letters. The structure of the network allows for weights to be pushed to earlier transitions without altering the ranking and cost of the accepted paths (double circles denote final states).

single iteration of the traditional algorithm for calculating the potential can be written in the form:

$$\mathbf{v}_{i+1} = \mathbf{v}_i \wedge \mathbf{A} \boxplus \mathbf{v}_i \tag{5}$$

where $\mathbf{v}_i$ is the potential vector for the i-th iteration, with $\mathbf{v}_0 = \boldsymbol{\rho}$, where $\boldsymbol{\rho}$ is the emission vector. By recursively substituting the values, we get that the final value of the potential vector is:

$$\mathbf{v}_\infty = \boldsymbol{\rho} \wedge \mathbf{A} \boxplus \boldsymbol{\rho} \wedge \mathbf{A}^2 \boxplus \boldsymbol{\rho} \wedge \ldots \wedge \mathbf{A}^n \boxplus \boldsymbol{\rho} \wedge \ldots = \Delta(\mathbf{A}) \boxplus \boldsymbol{\rho} \tag{6}$$

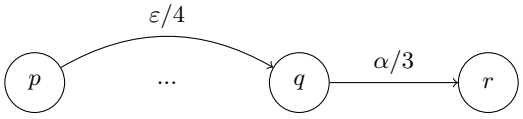
Claim 1. *The calculation of $\mathbf{v}_\infty = \Delta(\mathbf{A}) \boxplus \boldsymbol{\rho}$ in Eq. (6) is finite and $\Delta(\mathbf{A}) = \mathbf{I} \wedge \mathbf{A} \wedge \ldots \wedge \mathbf{A}^{n-1}$.*

The claim is proven by the fact that we have assumed that there aren't any cycles of negative length in the WFST, and such the shortest paths between every pair of states are finite. Having computed the potential vectors, we define four diagonal matrices that will be useful for updating the parameters of the WFST:

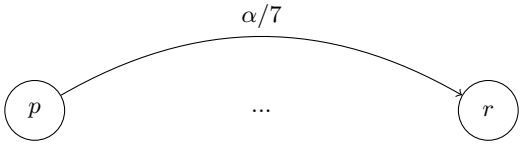
- The matrix $\boldsymbol{\Lambda}$ of the input weights, whose diagonal is the input weight vector $\boldsymbol{\lambda}$.
- The matrix $\mathbf{V}^+$ of the potentials, whose diagonal is the potential vector $\mathbf{v}_\infty$.
- The matrix $\mathbf{V}^-$ of the negative potentials, whose diagonal is the negative potential vector $-\mathbf{v}_\infty$.
- The matrix $\mathbf{P}$ of the emission weights, whose diagonal is the emission weight vector $\boldsymbol{\rho}$.

Having defined these matrices, the updated parameters of the WFST are as follows:

$$\boldsymbol{\lambda}' = \boldsymbol{\Lambda} \boxplus \mathbf{v}_\infty, \quad \boldsymbol{\rho}' = \mathbf{P} \boxplus (-\mathbf{v}_\infty), \quad \mathbf{A}' = \mathbf{V}^- \boxplus \mathbf{A} \boxplus \mathbf{V}^+ \tag{7}$$



(a) WFST before the epsilon removal operation.



(b) WFST after the epsilon removal operation.

Fig. 2: Epsilon transitions increase the total number of states and transitions of the WFST, reducing its efficiency. Removing states and replacing transitions, while maintaining the same accepted paths and costs, improves the run time of decoding algorithms.

Equation (7) models the weight pushing algorithm in tropical algebra. To highlight its function consider the five states of Fig. 1:

$$\mathbf{A} = \begin{bmatrix} \infty & 1 & 2 & \infty & \infty \\ \infty & \infty & \infty & 42 & \infty \\ \infty & \infty & \infty & \infty & 3 \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}, \boldsymbol{\lambda} = \begin{bmatrix} 0 \\ \infty \\ \infty \\ \infty \\ \infty \end{bmatrix}, \boldsymbol{\rho} = \begin{bmatrix} \infty \\ \infty \\ \infty \\ 2 \\ \infty \end{bmatrix} \tag{8}$$

Following the computations of Eq. (7) the updated matrices become:

$$\mathbf{A}' = \begin{bmatrix} \infty & 40 & 0 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & \infty & 0 \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}, \boldsymbol{\lambda}' = \begin{bmatrix} 5 \\ \infty \\ \infty \\ \infty \\ \infty \end{bmatrix}, \boldsymbol{\rho}' = \begin{bmatrix} \infty \\ \infty \\ \infty \\ 0 \\ \infty \end{bmatrix} \tag{9}$$

3.2. Epsilon removal

Epsilon removal is an algorithm that aims to reduce the number of states and transitions in the WFST, while maintaining its underlying structure, in order to reduce the running time of the Viterbi algorithm. To accomplish that, an effort is made to reduce epsilon transitions (meaning transitions with no input and output symbols).

The traditional algorithm for epsilon removal can be illustrated in Fig. 2. In essence, the algorithm computes, for each state p its epsilon closure, and then adds transitions from the state p to each state reachable from states in the epsilon closure. To model the traditional epsilon removal algorithm in tropical algebra we need to define two matrices:

- $\mathbf{\Sigma}_I$, which is the input symbol matrix and $(\Sigma_I)_{ij}$ contains the input symbol for the transition of the state i to state j.
- $\mathbf{\Sigma}_O$, which is the output symbol matrix and $(\Sigma_O)_{ij}$ contains the output symbol for the transition of the state i to state j.

Two remarks before we proceed to the modeling:

- We only consider as epsilon transitions ones where both the input and the output symbols are ε. This is a very common assumption, and a synchronization algorithm is executed in order to better match input and output ε.
- We assume that there can only be a single transition between two states, regardless of whether there exist transitions with different symbols or weights. While this might seem restrictive, in practice it isn't and can be circumvented.

We need to define another two matrices which make up the transition matrix $\mathbf{A}$ in order to model epsilon removal:

$$E_{ij} = \begin{cases} \alpha_{ij}, & \text{if } (\Sigma_I)_{ij} = (\Sigma_O)_{ij} = \varepsilon \\ \infty, & \text{otherwise} \end{cases}, \tag{10}$$

$$(A_\varepsilon)_{ij} = \begin{cases} \alpha_{ij}, & \text{if } (\Sigma_I)_{ij} = (\Sigma_O)_{ij} \neq \varepsilon \\ \infty, & \text{otherwise} \end{cases} \tag{11}$$

Essentially, we decompose matrix $\mathbf{A}$ using the matrices of Eq. (10). We can see that $\mathbf{A} = \mathbf{A}_\varepsilon \wedge \mathbf{E}$.

Claim 2. *Let* $\mathbf{E}$ *be the matrix defined in Eq.* (10)*. Then, the matrix*

$$\Gamma(\mathbf{E}) = \mathbf{E} \wedge \mathbf{E}^2 \wedge \ldots \wedge \mathbf{E}^n \wedge \ldots \tag{12}$$

is finite, equal to $\Gamma(\mathbf{E}) = \mathbf{E} \wedge \mathbf{E}^2 \wedge \ldots \wedge \mathbf{E}^{n-1}$*, and expresses the epsilon closure for all the states of the WFST.*

The claim is proven by the fact that an inherent assumption in WFSTs is that there aren't any cycles of negative weight (and thus the shortest distances are finite). In such a case, there aren't also any cycles of negative weight in the WFST of the epsilon transitions. Having computed epsilon closure of each state, the updated transition matrix and emission vector are simply the tropical addition between the previous values and the values from the epsilon closure. The new transition matrix $\mathbf{A}'$ takes the form:

$$\mathbf{A}' = \mathbf{A}_\varepsilon \wedge (\Gamma(\mathbf{E}) \boxplus \mathbf{A}_\varepsilon) = \Delta(\mathbf{E}) \boxplus \mathbf{A}_\varepsilon \tag{13}$$

whereas the new emission vector $\boldsymbol{\rho}'$ takes the form:

$$\boldsymbol{\rho}' = \boldsymbol{\rho} \wedge (\Gamma(\mathbf{E}) \boxplus \boldsymbol{\rho}) = \Delta(\mathbf{E}) \boxplus \boldsymbol{\rho} \tag{14}$$

Equations (13) and (14) model the epsilon removal algoirthm under the unified framework. We refer the reader to Fig. 2 for a visual explanation on how the algorithm removes redundant states without altering the accepted paths.

3.3. Viterbi Algorithm and Pruning

From a sequence of input symbols the Viterbi algorithm tries to estimate the sequence of states that has the highest probability. Formally, the Viterbi algorithm can be written in the following max-product form:

$$q_i(t) = \left(\max_j w_{ji} q_j(t-1) \right) \cdot b_i(\sigma_t) \tag{15}$$

where w_{ji} is the probability of transitioning from state j to state i, $b_i(\sigma_t)$ denotes the observation probability of the symbol σ_t at state i, and, finally, $q_i(t)$ is the maximum probability for that current state, calculated along the path from the previous states. In [19] we postulated that the Viterbi algorithm can be written in a closed matrix form in tropical algebra as:

$$\mathbf{x}(t) = \mathbf{P}(\sigma_t) \boxplus \mathbf{A}^T \boxplus \mathbf{x}(t-1) \tag{16}$$

where $\mathbf{x}(t) = -\log \mathbf{q}(t)$, $\mathbf{A} = -\log \mathbf{W}$, and $\mathbf{P}(\sigma_t)$ is a diagonal matrix whose diagonal is the vector $\mathbf{p}(\sigma_t) = -\log \mathbf{b}(\sigma_t)$.

The Viterbi pruning is a variant of the Viterbi algorithm that sacrifices the optimality of decoding in an effort to significantly speed up the process. Usually, pruning is based on one of the following criteria, or their combination:

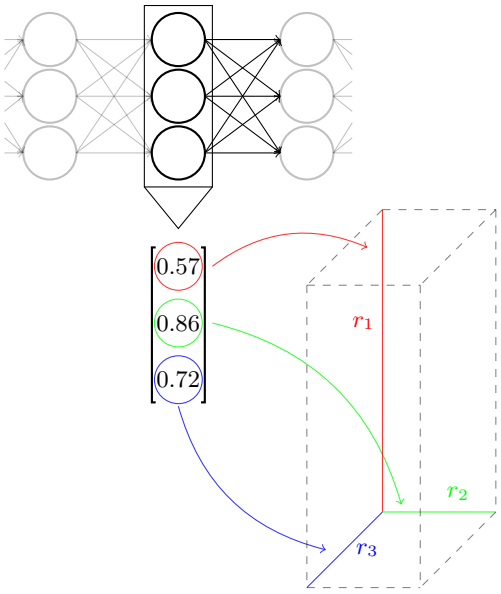


Fig. 3: At each step along the trellis, the state vector $\mathbf{x}(t)$ and the leniency vector $\boldsymbol{\eta}$ of Eq. (19) define a polytope. The three values of the vector $\mathbf{r}$ are shown with colors (see Eq. (20)).

- users determine a leniency parameter θ, and at each step only the paths that are at most θ from the optimal path survive.
- users determine a beam width κ, and at each step only the κ-best paths survive the pruning.

In [19] we modeled the Viterbi pruning in tropical algebra using *Cuninghame-Green's inverse* [11]. Therein it is proven that the negative elements of

$$\overline{\mathbf{y}} = \mathbf{X}^{\#}(t) \boxplus' \boldsymbol{\eta} \tag{17}$$

indicate the indices to be pruned, where $\boxplus'$ denotes the max-plus matrix multiplication. The matrix $\mathbf{X}(t)$ is a diagonal matrix whose diagonal is the state vector $\mathbf{x}(t)$, and $\mathbf{X}^{\#}(t) := -\mathbf{X}^{T}(t)$. Also, $\boldsymbol{\eta} = \theta + \frac{1}{2}\left(\mathbf{x}^{T}(t) \boxplus \mathbf{x}(t)\right) + \mathbf{0}$, where θ is the leniency parameter and $\mathbf{0}$ is a vector that consists of 0.

Let us consider a vector of variables $\mathbf{z}$ and bound it using

- the Viterbi update law of Eq. (15)

$$\mathbf{z} \geq \mathbf{b}, \quad \mathbf{b} = \mathbf{P}(\sigma_t) \boxplus \mathbf{A}^{T} \boxplus \mathbf{x}(t-1) \tag{18}$$

- the pruning vector of Eq. (17)

$$\mathbf{z} \leq \boldsymbol{\eta}, \quad \boldsymbol{\eta} = \theta + \frac{1}{2}\left(\mathbf{b}^{T} \boxplus \mathbf{b}\right) + \mathbf{0} \tag{19}$$

The combination of Eqs. (18) and (19) defines a tropical polytope for each step of the Viterbi algorithm. For each iteration two metrics are defined based on that polytope:

- a normalised volume metric ν:

$$\nu = -\frac{1}{|\mathrm{supp}(\mathbf{z})|} \sum_{i \in \mathrm{supp}(\mathbf{z})} \frac{\log r_i}{\log\left(\max \mathbf{r}\right)} \tag{20}$$

- a normalised entropy metric ε:

$$\varepsilon = -\frac{1}{|\mathrm{supp}(\mathbf{z})|} \sum_{i \in \mathrm{supp}(\mathbf{z})} -z_i(t) \cdot e^{-z_i(t)} \tag{21}$$

where $r_i = \eta - z_i$. Essentially, r_i is the degree to which each dimension satisfies the Viterbi constraints.

In Fig. 3 we highlight the components of the vector $\mathbf{r}$ in order to give a geometric explanation for the motivation of ν. We can see that r_i correspond to the individual lengths of the polytope, solidifying our characterisation of ν as a generalized volume.

4. ASPECTS OF GEOMETRY

We devote this section to the analysis of the Eqs. (20) and (21), and the motivation behind their definition. At every iteration of the Viterbi pruning consider the state vector $\mathbf{x}(t)$ along with the leniency vector $\boldsymbol{\eta}$ of Eq. (19). In unison, these vectors define a tropical polytope for each iteration of the algorithm. The indices of the state vector that satisfy the constraints imposed by the leniency vector act as the sides of this polytope, and the difference between the value of the leniency vector and the state vector constitute the vector $\mathbf{r}$ of Eq. (20). Figure 3 visualizes the polytope of each iteration and highlights the vector $\mathbf{r}$. Discussing the metrics further:

1) Consider the normalized volume of Eq. (20). The metric ν can offer a quantitative estimate of the solution space that the Viterbi pruning admits. Indeed, since the different r_i in Eq. (20) are normalized, this metric can provide a measure of how many paths the current choice of the leniency parameter θ allows to survive. Furthering that remark, we can monitor how ν evolves throughout iterations, and adapt the value of the leniency parameter θ in order to maintain a desired level of ν.

2) Consider the normalized entropy of Eq. (21). The metric ε can offer a qualitative estimate of the solution space that the Viterbi pruning admits. In information theory, entropy expresses the current degree of surprise incurred by the observation of a sample. In essence, if the sample abides by the existing modeling of the system, then it will have low entropy, as its value is in an expected range. However, if the sample has a significantly different value than those expected by the system's modeling, then the sample will have very high entropy, indicating that there may be an error in the modeling of the distribution.

By utilizing the above metrics we aim to reason about the solution space of the Viterbi pruning in two ways; a quantitative analysis of the relative size of the solution space, and a qualitative analysis of the likelihood of the paths of the solution space. Having such measures, we can examine how the solution space evolves over the execution of the Viterbi algorithm. Even more, we can introduce them to the design of the algorithm, so that the leniency parameter θ gets adapted to the needs of each iteration.

5. CONCLUSION

In this work we modeled algorithms that operate on WFSTs using tropical algebra and matrix operations on weighted lattices, unifying them under a common framework. First, we modeled the weight pushing algorithm by expressing the potential calculation as an instrumental matrix of tropical algebra. We then proceeded to model the epsilon removal algorithm by exploiting the $\min$-superposition of tropical algebra and expressing the epsilon closure as another important matrix in tropical algebra. Finally, we analyzed some geometrical aspects of the Viterbi pruning, elaborating on metrics that were defined on previous work. In future work we aim to explore the connection of these tropical matrix-based algorithms with the nonlinear vector spaces of weighted lattices and nonlinear spectral theory.

References

[1] M. Mohri, "Minimization Algorithms for Sequential Transducers," *Theoretical Computer Science*, vol. 234, pp. 177–201, 2000.

[2] M. Mohri, "Weighted Automata Algorithms," in *Handbook of Weighted Automata*, Springer, 2009, pp. 213–254.

[3] M. Mohri, F. Pereira, and M. Riley, "Weighted Finite-State Transducers in Speech Recognition," *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[4] A. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[5] G. D. Forney, "The Viterbi Algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[6] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[7] E. H. Kuo, "Viterbi Sequences and Polytopes," *Journal of Symbolic Computation*, vol. 41, pp. 151–163, 2006.

[8] L. Pachter and B. Sturmfels, "Tropical Geometry of Statistical Models," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 46, pp. 16 132–16 137, 2004.

[9] I. Simon, "On Semigroups of Matrices Over the Tropical Semiring," *Informatique Théorique et Applications*, vol. 28, no. 3-4, pp. 277–294, 1994.

[10] J.-E. Pin, "Tropical Semirings," in, J. Gunawardena, Ed. Idepotency, Cambridge University Press, 1998, pp. 50–69.

[11] R. Cuninghame-Green, *Minimax Algebra*, ser. Lecture Notes in Economics and Mathematical Systems. Springer, 1979.

[12] F. Baccelli, G. Cohen, G. J. Olsder, and J.-P. Quadrat, *Synchronization and Linearity: An Algebra for Discrete Event Systems*. John Wiley & Sons, 1993.

[13] S. Gaubert, *Methods and Applications of (max, +) Linear Algebra*, [Research Report], RR-3088, INRIA, 1997.

[14] P. Butkovič, *Max-linear Systems: Theory and Algorithms*. Springer, 2010.

[15] M. Gondran and M. Minoux, *Graphs, Dioids and Semirings: New Models and Algorithms*, ser. Operations Research/Computer Science Interfaces Series. Springer, 2008.

[16] P. Maragos, "Dynamical Systems on Weighted Lattices: General Theory," *Mathematics of Control, Signals, and Systems*, vol. 29, no. 21, pp. 1–33, 2017.

[17] B. Heidergott, G. J. Olsder, and J. van der Woude, *Max Plus at Work: Modeling and Analysis of Synchronized Systems: a Course on Max-Plus Algebra and Its Applications*. Princeton Univ. Press, 2006.

[18] V. Charisopoulos and P. Maragos, "Morphological Perceptrons: Geometry and Training Algorithms," in *Mathematical Morphology and Its Applications to Signal and Image Processing (ISMM 2017)*, ser. Lecture Notes in Computer Science, vol. 10225, J. Angulo, S. Velasco-Forero, and F. Meyer, Eds., Springer, 2017.

[19] E. Theodosis and P. Maragos, "Analysis of the Viterbi Algorithm Using Tropical Algebra and Geometry," in *Proceedings of IEEE International Workshop on Signal Processing Advances on Wireless Communications (SPAWC-2018)*, Kalamata, Greece, June 2018.

[20] P. Maragos, "Tropical Geometry, Mathematical Morphology and Weighted Lattices," in *Proceedings of International Symposium of Mathematical Morphology*, 2019.

[21] D. Maclagan and B. Sturmfels, *Introduction to Tropical Geometry*. American Mathematical Society, 2015.

[22] G. M. Ziegler, *Lectures on Polytopes*. Springer, 2012.

[23] S. Gaubert and R. D. Katz, "Minimal half-spaces and external representation of tropical polyhedra," *J. Algebr. Comb.*, vol. 33, pp. 325–348, 2011.

[24] H. Karloff, *Linear Programming*. Birkhäuser, 2008.

[25] V. Chvatal, *Linear Programming*. W. H. Freeman, 1983.

[26] D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming*. Springer, 2016.

[27] M. Sipser, *Introduction to the Theory of Computation*. Cengage Learning, 2012.