RAIRO-Theor. Inf. Appl. 56 (2022) 6
https://doi.org/10.1051/ita/2022005

RAIRO - Theoretical Informatics and Applications
www.rairo-ita.org

SYNTHESIZING FUZZY TREE AUTOMATA

Somaye Moghari*

Abstract. Fuzzy tree automata are mathematical devices for modeling and analyzing vaguely defined tree structures. The behavior of a fuzzy tree automaton generates a fuzzy tree language by mapping a set of regular trees on a ranked alphabet to fuzzy membership values. It calculates the membership grade of trees using a set of rules that process their structural characteristics. This paper deals with constructing fuzzy tree automata models that their behavior satisfies a set of given logical propositions (called properties) on the structure of trees. Our goal is uncertainty modeling by synthesizing fuzzy tree automata whose behavior is described by fuzzy linguistic variables. In this regard, we first provide several patterns and heuristic tricks and techniques for constructing fuzzy tree automata that satisfy simple properties. Then, we develop a method for modeling complex propositional formulas based on the conversion of a logical formula into a computation tree, as well as a step-by-step combination of models.

Mathematics Subject Classification. 68T37, 68Q45.

Received January 3, 2020. Accepted February 1, 2022.

1. Introduction

Studies in the behavior of complex systems led to the development of various computational models. Different types of modeling tools make available different kinds of analysis. One of the powerful computational models is automaton [12, 20, 38] which is an abstract machine with a set of transition rules. The transition rules enable automaton to move through a series of configurations. Each configuration is specified using a set of symbols called states. Some states are called final states, and if the automaton enters these states after processing, the input is accepted, and otherwise, it is rejected.

Modeling the uncertainty leads to the development of fuzzy automata [36] which is a class of automata with fuzzy transition rules and fuzzy configurations. A fuzzy automaton calculates a fuzzy grade of acceptance for each input term. As well, Fuzzy Tree Automata (FTA) [9, 13, 14, 31] deal with the logic of computation by processing tree structures and calculating the grade of acceptance for each input tree based on its structural characteristics. A Fuzzy Tree Language (FTL) is a fuzzy set of trees in which an FTA recognize them [4, 9, 32, 33, 36].

An FTA calculates the membership grade of trees to its language by a nonlinear mapping which is known as the behavior of the automaton [31]. The analysis of the behavior of automata [3, 32, 33] involves the analysis of states of automata and how these states change. On the other hand, modeling an FTL is the task of reasoning states and transition rules of an FTA which accepts it. In this paper, we refer to this task as the synthesis of

Keywords and phrases: Uncertainty modeling, synthesizing fuzzy tree automata, vaguely defined fuzzy tree languages.

Faculty of Mathematical Sciences, Shahrood University of Technology, Shahrood, Iran.

* Corresponding author: s.moghari@shahroodut.ac.ir

FTA. The modeling FTL could be expressed explicitly by formal terms [4] or implicitly by some specifications [23, 47]. Here, we consider the languages expressed by implicit descriptions through logical propositions.

The process of realizing implementation from specification (constructing model from specified properties) is a complicated task, especially when the properties are expressed vaguely and imprecisely. This paper provides a series of practical techniques in the form of patterns that can be used in approximating an FTA to model an FTL expressed by linguistic variables [49, 50]. Here, we deal with this problem in three main steps. The first step is modeling atomic propositions, which are simple statements or assertions on trees. We provide several patterns for modeling atomic propositions that are true or false (nonfuzzy atomic propositions), as well as those that are not precisely defined and have a degree of truth (fuzzy atomic propositions). The second step includes modeling the combination of two atomic propositions by logical connectives $\vee$, $\wedge$, and $\neg$. In this step, we improve the model by defining multi-indices states, refactoring transition rules, and developing final state membership functions by logical connectives to show how combining two models can support the combination of the corresponding properties. The final step is modeling a set of more complex properties that express an FTL. To construct an FTA corresponding to all of these properties, we first organize them into an expression tree [42] and then employ a bottom-up construction method.

The modeling patterns and techniques presented in this article are useful for modeling in three ways. First, each pattern is related to a family of properties and can be used in similar modeling. Second, the proposed multi-indexing techniques for FTA integration can generally be used for automata combinations. Third, a systematic approach to the development of complex models is provided.

This paper is organized as follows. Section 2 addresses some of the works related to this research, and Section 3 includes the basic concepts about fuzzy sets, ranked alphabets, and FTA. The proposed methods and techniques for modeling some patterns of FTL are presented in Section 4. Finally, Section 5 summarizes the achievements of this research.

2. Related work

Usually, synthesis and analysis of systems by computational models is associated with uncertainty modeling, and approximate reasoning [2, 10]. Fuzzy logic is capable of overcoming the vagueness, imprecision and absence of concrete data for generating robust models to drive decisions from uncertainties [41].

Researchers in various fields have defined and developed different fuzzy models to perform computation and analysis. Fuzzy automata are computational models that can describe practical processes [27]. Ying and Lin in [46] have used fuzzy automata as the model of fuzzy discrete event systems and developed online learning algorithms to adjust its parameters. They applied stochastic gradient descent for optimizing the transition matrix. Also, Du and Zhu's researches on the analysis of social networks by fuzzy automata led to defining fuzzy relational structure and deducing knowledge by computing lower and upper approximations based on bisimulation [8]. The bisimulation relation models equivalence between states of nondeterministic automata [45]. The equivalence relation between states of an automaton is done to minimize the automaton, and the equivalence relation between states of two automata is usually done for comparing their behaviors [13, 14, 32].

The challenge of modeling hybrid automata is addressed in [28]. Their idea is to construct the model by processing the set of unlabeled data belonging to the language of the model. They clustered the data to obtain language subclasses, each labeled with a state symbol, and then modeled the transition rules by searching for binary classification boundaries. Soto *et al.* [40] focused on creating models from experimental data, as well. They synthesized linear hybrid automata and developed an adaptive method for optimizing the model to minimize the number of modes. Also, they developed another adaptive algorithm to optimize the model by increasing its precision. There exist some other works on the training of hybrid automata by analysis of input/output traces and machine learning techniques (see [25, 29]).

Cellular automata (CA) is a computational model for systems that changes occur at any point (cell) based on the status or history of its neighboring points [6, 30]. A CA consists of a collection of cells arranged in a multi-dimensional space and a dynamical rule (transition function) that updates their configuration synchronously. The transition function calculates the state of each cell based on the state of its neighbors, where these simple

local interactions and computations between cells, form a complex global behavior [30]. The set of transition rules of a CA is a type of knowledge that Li and Yeh [26] applied data mining technique for reconstructing it. They constructed a CA model for geographical phenomena by processing a series of spatial data, including the layers of urban development, proximity variables, neighborhood conditions, and physical attributes. Also, He *et al.* [18] used deep learning techniques to extract transition rules of CA and constructed a prediction model of urban expansion pattern. They trained a convolution neural network using geographical data of the study area. Roodposhti *et al.* [39] provided a dictionary of trusted rules (called DoTRules) to calculate transition potential in CA models of land-use/cover change. DoTRules supports generating transparent transition rules and quantifies uncertainty by estimating the frequency and entropy of major land-use transitions.

The works mentioned above can be categorized in the area of active learning of automata models [21] which is inferring models from observations. This paper aims to provide some synthesis patterns for FTA models corresponding to FTL that are not described by formal terms but fuzzy linguistic variables. Fuzzy linguistic variables are powerful tools for the approximate characterization of concepts that are not well-defined to be described in quantitative terms [22, 49, 50]. It can be used in fuzzy modeling of requirements to present a method for fuzzy intelligent requirement engineering from natural language to computer-aided design models [11]. Also, linguistic variables are employed to develop analytical tools for data mining and time series prediction [15]. Of course, linguistic variables and fuzzy membership functions require systematic methods to define and evaluate [37].

3. Preliminaries

3.1. Fuzzy sets

Basic concepts will be used as Zadeh's fuzzy logic and fuzzy set theory [16, 48, 51]. Throughout the article, we use ℓ as the unit interval $[0,1] \in \mathbb{R}$.

Definition 3.1. Let X be a collection of elements or objects. A fuzzy set A in X is a set of ordered pairs

$$A = \{(x, \mu_A(x)) \mid x \in X\},$$

where $\mu_A(x)$ is called the membership function or grade of membership of x in A and (generally) lie in ℓ. We denote by $\widetilde{X}$ the set of all fuzzy sets on X. Also, a fuzzy number refers to the fuzzy set representing the possible values for a number.

3.2. Ranked alphabet and fuzzy tree Automata

Our definitions in this section, are borrowed from [7, 32, 33].

The set of natural numbers (including zero) is denoted by $\mathbb{N}$. A Σ-*alphabet* is a finite and nonempty set of symbols. A *ranked alphabet* is a couple $(\Sigma, Arity)$, which is the disjoint union of sets of n-*ary* symbols $\Sigma_n = \{\sigma \mid Arity(\sigma) = n\}$ for all $n \in \mathbb{N}$. The set of symbols of arity 0 are called constants.

Let Q be a set of constants called *variables*. The set $T_\Sigma(Q)$ of $\Sigma - trees$ indexed by Q is inductively defined to be the smallest set such that $Q \subseteq T_\Sigma(Q)$ and $\sigma(t_1, \ldots, t_n) \subseteq T_\Sigma(Q)$ for every $\sigma \in \Sigma_n$ and $t_1, \ldots, t_n \in T_\Sigma(Q)$. We write T_Σ for $T_\Sigma(\phi)$. The *size* and the *height* of a tree $t \in T_\Sigma(Q)$ denoted by $\mathbb{S}(t)$ and $\mathbb{H}(t)$ respectively, are inductively defined by

- $\mathbb{S}(t) = 0$ and $\mathbb{H}(t) = 0$ for $t \in Q$,
- $\mathbb{S}(t) = 1$ and $\mathbb{H}(t) = 1$ for $t \in \Sigma_0$,
- $\mathbb{S}(t) = 1 + \mathbb{S}(t_1) + \ldots + \mathbb{S}(t_n)$ and $\mathbb{H}(t) = 1 + max\{\mathbb{S}(t_1), \ldots \mathbb{S}(t_n)\}$ for $n \geq 1$, $t = \sigma(t_1, \ldots, t_n)$ and $t_1, \ldots, t_n \in T_\Sigma(Q)$.

Definition 3.2. A *fuzzy tree automaton* is a system $M = (\Sigma, Q, \Gamma, \delta, \ell, \rho, \beta)$, where:

1. Σ is a finite set of ranked alphabets called *input symbols*.

2. Q is a finite set of symbols called *states*.
3. $\Gamma : Q \to \ell$ is a fuzzy set on Q and called the set of *final states*.
4. $\delta = \{\delta_\sigma : Q^n \times Q \times \Sigma_n \to \ell \mid \sigma \in \Sigma_n,\ n \in \mathbb{N}\}$ is a finite set called *transition rules*.
5. $\rho : T_\Sigma(Q) \times Q \to \ell$ is called the run map of FTA M, and defined by induction on structure of $t \in T_\Sigma(Q)$:
   (a) If $t = \sigma \in \Sigma_0$, then $\rho(t)(q) = \delta(q, \sigma)$, for all $q \in Q$,
   (b) If $t = \sigma(t_1, \ldots, t_n)$ for some $\sigma \in \Sigma_n$ and $t_1, \ldots, t_n \in T_\Sigma$, then

$$\rho(t)(q) = \bigvee_{q_1,\ldots,q_n \ \in Q} \Big(\delta(q_1, \ldots, q_n, q, \sigma) \wedge \bigwedge_{i=1}^{n} \rho(t_i)(q_i) \Big).$$

6. $\beta : T_\Sigma \to \ell$ is a fuzzy set on the set of trees $t \in T_\Sigma$, called behavior of FTA M, and defined by:

$$\beta(t) = \bigvee_{q \in Q} \Big(\rho(t)(q) \wedge \Gamma(q) \Big).$$

An FTA $M = (\Sigma, Q, \Gamma, \delta, \ell, \rho, \beta)$ *accepts(recognizes)* a tree $t \in T_\Sigma$ if and only if $\beta(t) > 0$. Also, the set of all trees accepted by M is known as FTL $L(M)$, where $\mu_{L(M)}(t) = \beta(t)$.

Example 3.3. Let FTA $M = (\Sigma, Q, \Gamma, \delta, \ell, \rho, \beta)$be defined as follows.

- $\Sigma = \{a, f\}$; $a \in \Sigma_0$, $f \in \Sigma_2$.
- $Q = \{q_1, q_2\}$.
- $\Gamma(q_1) = 0.8$, $\Gamma(q_2) = 1$.
- $\delta(q_1, a) = 0.9$, $\delta(q_1, q_1, q_1, f) = 0.5$, $\delta(q_1, q_2, q_1, f) = 0.6$,
  $\delta(q_2, a) = 0.6$, $\delta(q_2, q_1, q_2, f) = 0.3$, $\delta(q_2, q_2, q_2, f) = 0.4$.

Now, there exists two run maps for tree $t_1 = a$:

$$\rho(t_1)(q_1) = \delta(q_1, a) = 0.9$$

$$\rho(t_1)(q_2) = \delta(q_2, a) = 0.6$$

Also, the behavior of FTA M on tree $t_1 = a$ is calculated by:

$$\beta(t_1) = \Big(\rho(t_1)(q_1) \wedge \Gamma(q_1) \Big) \vee \Big(\rho(t_1)(q_2) \wedge \Gamma(q_2) \Big) = 0.8 \vee 0.6 = 0.8.$$

So, t_1 blongs to FTL $L(M)$ with membership degree 0.8.
As well, there exists two run maps for tree $t_2 = f(a, a)$:

$$\begin{aligned} \rho(t_2)(q_1) &= \Big(\delta(q_1, q_1, q_1, f) \wedge \rho(a)(q_1) \wedge \rho(a)(q_1) \Big) \vee \\ &\quad \Big(\delta(q_1, q_2, q_1, f) \wedge \rho(a)(q_1) \wedge \rho(a)(q_2) \Big) \\ &= \Big(0.5 \wedge 0.9 \wedge 0.9 \Big) \vee \Big(0.6 \wedge 0.9 \wedge 0.6 \Big) = 0.6 \end{aligned}$$

$$\rho(t_2)(q_2) = \Big(\delta(q_2,q_1,q_2,f) \wedge \rho(a)(q_2) \wedge \rho(a)(q_1)\Big) \vee \Big(\delta(q_2,q_2,q_2,f) \wedge \rho(a)(q_2) \wedge \rho(a)(q_2)\Big) = \Big(0.3 \wedge 0.6 \wedge 0.9\Big) \vee \Big(0.4 \wedge 0.6 \wedge 0.6\Big) = 0.4$$

So, the behavior of FTA M on tree $t_2 = f(a,a)$ is calculated by:

$$\beta(t_2) = \Big(\rho(t_2)(q_1) \wedge \Gamma(q_1)\Big) \vee \Big(\rho(t_2)(q_2) \wedge \Gamma(q_2)\Big) = 0.6 \vee 0.4 = 0.6.$$

Therefore, t_2 blongs to FTL $L(M)$ with membership degree 0.6.

4. Synthesizing FTA corresponding to a linguistically described FTL

Let an FTL be described by a set of logical statements, some of which may be ambiguous. It is notable that the logical statement is called property in general, and is called atomic proposition when it is simple and uncomposed [5]. Our goal is to define an FTA that recognizes this language as accurately as possible. In this regard, we have provided a number of patterns, each of which includes three main parts; a generic title, a property (FTL) and the corresponding model (FTA). The patterns presented in this section are defined based on some structural patterns on XML data [1, 17, 19, 24, 34, 35, 43, 44].

Throughout this section, the constructing FTA is $M = (\Sigma, Q, \Gamma, \delta, \ell, \rho, \beta)$ where the properties are defined on $\Sigma = \{a, b, g, f\}$, $a, b \in \Sigma_0$, $g \in \Sigma_1$, $f \in \Sigma_2$, and we have to define Q, Γ and δ to complete the model.

4.1. Modeling atomic propositions

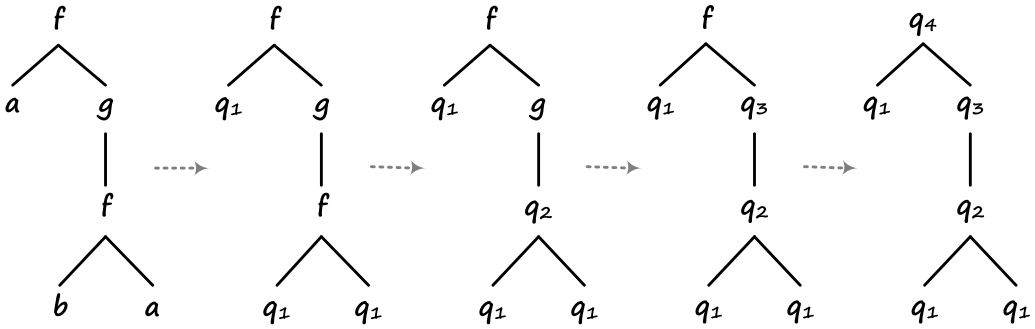
An atomic proposition is a sentence that has a truth value and cannot be decomposed into simpler sentences. In this section, we present some patterns of atomic propositions and the required tricks for modeling them by FTA. The patterns include some fuzzy/nonfuzzy restrictions on height and size of trees, fuzzy/nonfuzzy symbol counting, symbol detection, and tree structure pattern recognition.

Pattern 4.1. Threshold on the height of trees

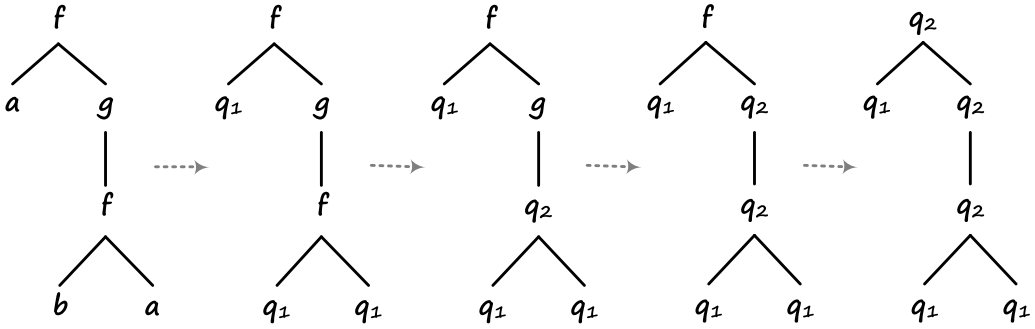
Property: "The height of the tree is less than H".

Model: $Q = \{q_1, \ldots, q_H\}$, $\Gamma(q_H) = 0$ and $\Gamma(q_h) = 1$ for $h < H$.
$\delta(q_1, a) = 1$, $\delta(q_1, b) = 1$, $\delta(q_i, q_{min\{i+1,H\}}, g) = 1$ and
$\delta(q_i, q_j, q_{min\{max\{i,j\}+1,H\}}, f) = 1$ for $i, j \leq H$.

In Model 4.1, the processing of constant symbols results in state q_1 and for nodes that have children, given that i is the largest state index of the children, if $i < H$ then processing will result in q_{i+1} and otherwise it will be q_H. The processing of each tree is bottom-up (from leaves to root), and whenever a node has no children or all of its children have been processed, it can be processed. Here, q_h represents the state of nodes at the root of trees whose height is h. Of course, for trees whose height is greater than or equal to H the desired condition is violated and for all of them we consider state q_H with $\Gamma(q_H) = 0$. Figure 1 shows an example of step-by-step running of two different FTA derived from Model 4.1 to process the same tree structure. In Figure 1(a), states q_1 to q_4 are generated in steps 1 to 4, respectively, but in Figure 1(b), after entering state q_2, the desired condition is violated and no new states are generated at parent nodes and this state is repeated until the root of the tree.



a) Running the model corresponding to any H ≥ 4



b) Running the model corresponding to H = 2

FIGURE 1. Steps of running two different FTA corresponding to Model 4.1 on tree $f(a, g(f(b, a)))$.

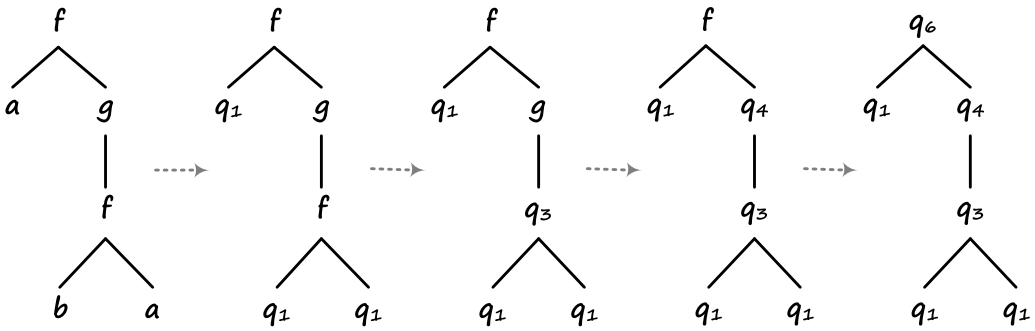
Pattern 4.2. Fuzzy threshold on the height of trees

Property: "The height of the tree is approximately H".

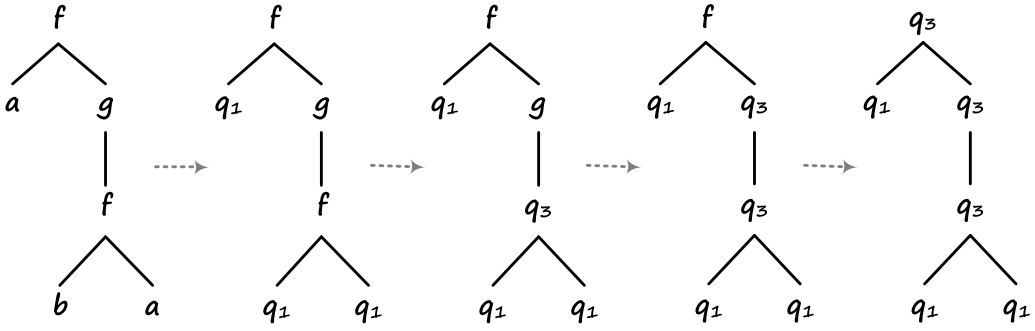
Model: Let $\hat{\mathrm{H}}$ is the fuzzy number denoting approximately H and $U \in \mathbb{N}$ is the smallest number that for any $x \in \mathbb{N}$ it holds $x \geq U \Rightarrow \mu_{\hat{\mathrm{H}}}(x) = 0$.
$Q = \{q_1, \ldots, q_U\}$ and $\Gamma(q_h)$ is compatible with $\mu_{\hat{\mathrm{H}}}(h)$ for $1 \leq h \leq U$.
$\delta(q_1, a) = 1$, $\delta(q_1, b) = 1$, $\delta(q_i, q_{min\{i+1,U\}}, g) = 1$ and
$\delta(q_i, q_j, q_{min\{max\{i,j\}+1,U\}}, f) = 1$ for $i, j \leq U$.

Model 4.2 is similar to Model 4.1 except the final states Γ. Here the value of approximately H is considered as the fuzzy number $\hat{\mathrm{H}}$ and state symbol q_h is intended for the root of trees of height h. So, the membership value $\Gamma(q_h)$ is set proportional to the membership value $\mu_{\hat{\mathrm{H}}}(h)$. For example, let $H = 7$ and M be an FTA constructed based on Model 4.2, where "approximately 7" is defined by triangular fuzzy number $\hat{7} = T(3, 7, 11)$. Now, for tree $t = f(a, g(f(b, a)))$ shown in Figure 1, we have $L_M(t) = 0.25$ because processing of t with FTA M leads to state q_4 and $\mu_{\hat{7}}(4) = 0.25$.

Pattern 4.3. Threshold on the size of trees



a) Running the model corresponding to any S ≥ 5



b) Running the model corresponding to S = 2

FIGURE 2. Steps of running two different FTA corresponding to Model 4.3 on tree $f(a, g(f(b, a)))$.

Property: "The size of the tree is greater than S".

Model: $Q = \{q_1, \ldots, q_{S+1}\}$, $\Gamma(q_{S+1}) = 1$ and $\Gamma(q_i) = 0$ for $i \leq S$.
$\delta(q_1, a) = 1$, $\delta(q_1, b) = 1$, $\delta(q_i, q_{min\{i,S\}+1}, g) = 1$ and
$\delta(q_i, q_j, q_{min\{i+j,S\}+1}, f) = 1$ for $1 \leq i, j \leq S + 1$.

In Model 4.3, q_i is the state symbol for the root of trees with size i, where $i \leq S$. Also, for trees which their sizes are greater than S the desired condition is reached and we consider state q_{S+1} with $\Gamma(q_{S+1}) = 1$ for them. Figure 2 shows an example of step by step running of two different FTA derived from Model 4.3 to process the same tree structure. In Figure 2(a), $S \geq 5$ which implies $\{q_1, q_2, \ldots, q_6\} \subseteq Q$. So, states q_1, q_3, q_4 and q_6 are generated in steps 1 to 4, respectively. In Figure 2(b), $Q = \{q_1, q_2, q_3\}$ and after entering state q_3, the desired condition is satisfied and no new state is generated at parent nodes and this state is repeated until the root of the tree.

Pattern 4.4. Constraint with iterative pattern on height

Property: "The height of the tree is a multiple of M".

Model: $Q = \{q_0, \ldots, q_{M-1}\}$, $\Gamma(q_0) = 1$ and $\Gamma(q_i) = 0$ for $i > 0$.
$\delta(q_1, a) = 1$, $\delta(q_1, b) = 1$, $\delta(q_i, q_{(i+1) \ mod \ M}, g) = 1$ and
$\delta(q_i, q_j, q_{(max\{i,j\}+1) \ mod \ M}, f) = 1$ for $i, j < M$.

In Model 4.4, indices of state symbols represent the remainder after division of the height of trees by M. Since the trees are processed bottom-up, at each step, the index of reached states is increased by 1 until state q_{M-1} is reached; then the index value is reset.

Pattern 4.5. Restriction that leads to a non-regular tree language

Property: "The size of the tree is a prime number".

Model: The mapping between the size (or height) of trees and the set of prime numbers is not a regular tree language and can not be recognized by an FTA (Example 1.2.2 in [7]).

According to the Pumping Lemma for recognizable tree languages [7] it can be shown that Model 4.5 is not corresponding to a recognizable FTL. Understanding the patterns that lead to unrecognizable tree languages in combination with the closure properties of recognizable tree languages enables us to deduce whether the language is recognizable or not. Detecting the class of tree language of an atomic proposition is useful, especially when we are modeling a language described by more complex properties, because it can prevent a large number of calculations resulting from modeling and combining them. However, a modeling tool can have separate libraries for patterns related to different language classes.

Pattern 4.6. Semi-atomic constraint with iterative pattern on the size of trees

Property: "The size of the tree is a multiple of M and N".

Model: Let K is the least common multiple of M and N.
$Q = \{q_0, \ldots, q_{K-1}\}$, $\Gamma(q_0) = 1$ and $\Gamma(q_i) = 0$ for $i > 0$.
$\delta(q_1, a) = 1$, $\delta(q_1, b) = 1$, $\delta(q_i, q_{(i+1) \ mod \ K}, g) = 1$ and
$\delta(q_i, q_j, q_{(max\{i,j\}+1) \ mod \ K}, f) = 1$ for $i, j < K$.

The modeling property in Pattern 4.6 is not atomic because it can be broken down into properties "*The size of the tree is a multiple of M*" and "*The size of the tree is a multiple of N*". However, it is equivalent to the atomic property "*The size of the tree is a multiple of K*" where K is the least common multiple of M and N. So, we call it a "semi-atomic property". Note that detecting and optimizing semi-atomic properties can reduce modeling costs by reducing time complexity and memory requirements.

Pattern 4.7. Symbol detection

Property: "The tree has a node labeled **g**".

Model: $Q = \{q_0, q_1\}$, $\Gamma(q_0) = 0$ and $\Gamma(q_1) = 1$.
$\delta(q_0, a) = 1$, $\delta(q_0, b) = 1$, $\delta(q_i, q_1, g) = 1$ and $\delta(q_i, q_j, q_{max\{i,j\}}, f) = 1$
for $i, j \in \{0, 1\}$.

In Pattern 4.7, two state symbols q_0 and q_1 indicate the absence and presence of the desired symbol in the processing tree structure. Therefore, state symbol q_0 is labeled on nodes until $g \in \Sigma$ is scanned, and then, all

remaining nodes will result in state q_1. Note that the existence of one symbol can be extended to the existence of multiple symbols, and the same pattern can be used for modeling.

Pattern 4.8. Symbol counting

Property: "The tree has at least M nodes labeled **b**".

Model: $Q = \{q_0, \ldots, q_M\}$, $\Gamma(q_M) = 1$ and $\Gamma(q_i) = 0$ for $0 \leq i < M$.
$\delta(q_0, a) = 1$, $\delta(q_1, b) = 1$, $\delta(q_i, q_i, g) = 1$ and
$\delta(q_i, q_j, q_{min\{i+j,M\}}, f) = 1$ for $i, j \leq M$.

Pattern 4.8 is a generalization of Pattern 4.7 such that the desired symbol is counted, and if at least M samples are met, the processing tree structure is accepted. It is clear that if $M = 1$, these two patterns are equivalent. In other words, the symbol detection pattern is a special case of the symbol counting pattern, where the threshold of counting symbols is 1. Note that counting one symbol can be extended to counting several symbols simultaneously, and the same pattern can be used for modeling.

Pattern 4.9. Pattern recognition

Property: "The tree contains sub-structure **t=f(b,f(g(-),a))**"

Model: $Q = \{q_1, \ldots, q_6\}$, $\Gamma(q_5) = 1$ and $\Gamma(q_i) = 0$ for $i \neq 5$.
$\delta(q_1, a) = 1$, $\delta(q_2, b) = 1$, $\delta(q_k, q_3, g) = 1$, $\delta(q_5, q_5, g) = 1$, $\delta(q_3, q_1, q_4, f) = 1$,
$\delta(q_2, q_4, q_5, f) = 1$, $\delta(q_i, q_j, q_6, f) = 1$, $\delta(q_m, q_5, q_5, f) = 1$ and $\delta(q_5, q_m, q_5, f) = 1$
for $i, j, k \neq 5$, $(i, j) \notin \{(3, 1), (2, 4)\}$ and $m \leq 6$.

In Model 4.9, the input pattern has 5 nodes, and the FTA uses state symbols $\{q_1, \ldots, q_5\}$ for step by step labeling of the input tree during the pattern recognition. Also, state symbol q_6 is corresponding to situation that a node violates all sub-structures of the pattern while pattern t is not met. Figure 3 shows the steps of running the FTA derived from Model 4.9 on tree structure $t = f(b, f(g(t'), a))$ where $t' \in T_\Sigma$. Here, we used a set of labels to describe the meaning of states:

- A : The current node is a.
- B : The current node is b.
- G : The current node is g and pattern t has not been found yet.
- F : The current pattern is $f(g(-), a)$ and pattern t is not found yet.
- T : Pattern t is found.

According to Figure 3, if pattern t is found in the processed tree, the model remains in state q_5, and the processing continues in the same way until the root of the tree. Also, if pattern t is not found, it is detected step by step according to all subtree decompositions.

Pattern 4.10. Comparing the number of symbols

Property: "The number of **a** and **b** is equal in all subtrees with root **f**".

Model: $Q = \{q_0, \ldots, q_3\}$, $\Gamma(q_3) = 0$, $\Gamma(q_i) = 1$ for $i \leq 2$.
$\delta(q_1, a) = 1$, $\delta(q_2, b) = 1$, $\delta(q, q, g) = 1$, $\delta(q_i, q_j, q_0, f) = 1$, $\delta(q_m, q_n, q_3, f) = 1$
for every $q \in Q$, $(i, j) \in \{(0, 0), (1, 2), (2, 1)\}$ and $(m, n) \notin \{(0, 0), (1, 2), (2, 1)\}$.

State symbol q_3 in modeling part of Pattern 4.10 is a trap state; because if the condition is violated at a node, it leads to q_3 and this state symbol will be propagated to all parent nodes, and it causes the tree to be rejected.

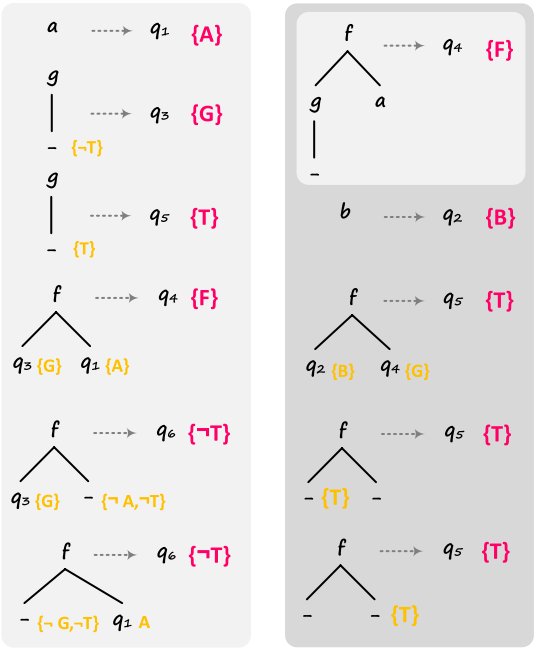


FIGURE 3. Steps of detecting pattern $t = f(b, f(g(-), a))$ by Model 4.9.

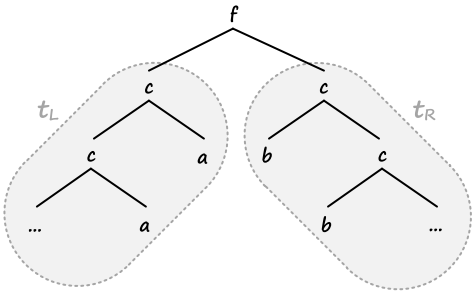


FIGURE 4. There is no FTA that can compare the number of a in subtree t_L with the number of b in subtree t_R.

When the pattern of the processing tree is a, $g(a)$ and $g(g(\dots g(a)))$, it leads to q_1 and for patterns b, $g(b)$ and $g(g(\dots g(b)))$, it leads to q_2. So, if a tree (subtree) leads to q_1/q_2, it has only one a/b, respectively. Consequently, for each node f that its children are processed, there are only three acceptable cases $f(q_0, q_0)$, $f(q_1, q_2)$ and $f(q_2, q_1)$, which all leads to q_0. Symbol state q_0 means that the number of symbols **a** and **b** are equal in the corresponding tree (subtree). Other cases (*e.g.*, $f(q_0, q_1)$, $f(q_0, q_2)$ and $f(q_1, q_1)$) are not acceptable because they are the result of trees (*e.g.*, $f(f(a, b), a)$, $f(f(b, a), g(g(b)))$ and $f(g(a), a)$) that violate the condition.

The FTL presented in Pattern 4.10 is not expandable to support adding a new alphabet of rank 2 or more into Σ. For example, if new symbol $c \in \Sigma_2$ is added to Σ, the pattern will be a context free tree language and can not be presented by a finite tree automaton (Fig. 4). Context free tree languages can be recognized by pushdown tree automata [7].

Pattern 4.11. Special context-free symbol pattern

Property: "The number of **a** and **b** is equal if the root of the tree is **f**".

Model: This property could not be modeled with a finite FTA. Pushdown tree automata and tree automata with infinite number of states are two classes of tree automata that can recognize it.

Proposition 4.12. *Pattern 4.11 is not a recognizable tree language.*

Proof. The corresponding tree language is
$L = \{g^i(a) \mid i \geq 0\} \cup \{g^i(b) \mid i \geq 0\} \cup \{f(t_1, t_2) \mid n_a(t_1) + n_a(t_2) = n_b(t_1) + n_b(t_2)\}$.
Suppose that L is recognizable by FTA M having k state symbols. Now, let $t = f(t_1, t_2)$ with $t_1 = \underbrace{f(f(\ldots f}_{k+1}(a, a)\ldots, a), a)$ and $t_2 = \underbrace{f(f(\ldots f}_{k+1}(b, b)\ldots, b), b)$. Since $t \in L$, there is a successful run of M on t. As k is the cardinality of the state set of M, there are two distinct nodes f along t_1 labeled with the same state. Therefore, one could cut the first branch between these two positions leading to a tree $t' = f(t'_1, t_2)$ with $t'_1 = \underbrace{f(f(\ldots f}_{j}(a, a)\ldots, a), a)$, where $j < k + 1$ and a successful run of M exists for t'. This is a contradiction with $L(M) = L$. □

Pattern 4.13. Fuzzy comparison of the symbols

Property: "The number of **a** and **b** is almost equal in all subtrees with root **f**".

Model: Let $\hat{Z}$ is a symmetric fuzzy number corresponding to *"almost zero"*, and $L, R \in \mathbb{Z}$ are bounds of $\hat{Z}$ such that for every $x \in \mathbb{Z}$ it holds $\mu_{\hat{Z}}(x) > 0 \Leftrightarrow x \in (L, R)$. Also, consider mapping $\gamma : \mathbb{Z} \to [L, R]$ defined by

$$\gamma(x) = \begin{cases} L & x \leq L \\ x & L < x < R \\ R & x \geq R \end{cases}$$

Now, the corresponding model is:
$Q = \{q_L, \ldots, q_R\}$ and $\Gamma(q_i) = \mu_{\hat{Z}}(i)$ for $L \leq i \leq R$.
$\delta(q_1, a) = 1$, $\delta(q_{-1}, b) = 1$, $\delta(q_k, q_k, g) = 1$, $\delta(q_i, q_j, q_{\gamma(i+j)}, f) = 1$,
$\delta(q_L, q_k, q_L, f) = 1$, $\delta(q_k, q_L, q_L, f) = 1$, $\delta(q_k, q_R, q_R, f) = 1$ and
$\delta(q_R, q_k, q_R, f) = 1$, where $k \in [L, R]$ and $i, j \in (L, R)$.

In the modeling presented in Pattern 4.13, for $i \in (L, R)$, each state symbol q_i specifies the state of the nodes where the number of a's minus the number of b's in the corresponding subtree is i. Also, q_L and q_R are two trap state symbols indicating that the condition is violated in the current subtree or at least one of its subtrees.

4.2. Modeling properties with logical connectives

In this section, we first provide three sample patterns that use multiple indexing of state symbols and more advanced membership functions for final states to model the fuzzy properties that are the combination of some atomic propositions by logical connectives $\vee$, $\wedge$, and $\neg$. Then, we present a general method for combining different FTA to model more complex properties.

Pattern 4.14. (Threshold on the size of trees) $\wedge$ (Fuzzy threshold on the height of trees)

Property: "The size of the tree is less than S **and** its height is about H".

Model: Let $\hat{\mathrm{H}}$ is the fuzzy number corresponding to the term about H, and $U \in \mathbb{Z}$ is the upper bound of $\hat{\mathrm{H}}$ such that for every $x \in \mathbb{Z}$ it holds $x \geq U \Rightarrow \mu_{\hat{\mathrm{H}}}(x) = 0$.
$Q = \{q_{(i,j)} \mid 1 \leq i \leq S, 1 \leq j \leq U\}$,
$\Gamma(q_{(S,j)}) = 0$ and $\Gamma(q_{(i,j)}) = \mu_{\hat{\mathrm{H}}}(j)$ for $i < S$ and $j \leq U$.
Note that this definition of final states is resulted from $\Gamma(q_i) \wedge \Gamma(q_j)$ where $\Gamma(q_i)$ and $\Gamma(q_j)$ correspond to atomic propositions *threshold for size* and *fuzzy threshold for height*, respectively.
$\delta(q_{(1,1)}, a) = 1$, $\delta(q_{(1,1)}, b) = 1$, $\delta(q_{(i,j)}, q_{(min\{i+1,S\},min\{j+1,U\})}, g) = 1$ and
$\delta(q_{(i,i')}, q_{(j,j')}, q_{(min\{i+i'+1,S\},min\{max\{i,j\}+1,U\})}, f) = 1$ for $i, i' \leq S$ and
$j, j' \leq U$.

Pattern 4.15. (Threshold on the size of trees) $\vee$ (Fuzzy threshold on the height of trees)

Property: "The size of tree must be less than S **or** its height must be about H".

Model: It is sufficient that the set of final states in the modeling of Pattern 4.14 be redefined as follows
$\Gamma(q_{(S,U)}) = 0$ and $\Gamma(q_{(i,j)}) = \mu_{\hat{\mathrm{H}}}(j)$ for $(i,j) \neq (S,U)$.
This definition of final states, is resulted from $\Gamma(q_i) \vee \Gamma(q_j)$ where $\Gamma(q_i)$ and $\Gamma(q_j)$ correspond to atomic propositions *threshold for size* and *fuzzy threshold for height*, respectively.

The set of transition rules of modelings in Patterns 4.14 and 4.15 are the same, where both two FTA assign the state symbol $q_{(i,j)}$ to each tree so that i presents its size (or violation of the related property) and j presents its height (or violation of the related property).

Pattern 4.16. Fuzzy symbol counting pattern associated with $\neg$

Property: "The number of nodes labeled **b** in the tree is **not** significant".

Model: Consider $\hat{\mathrm{S}}$ is the fuzzy number corresponding to significant and $U \in \mathbb{N}$ is a natural number that $\mu_{\hat{\mathrm{S}}}(i) = 1$ for every $i \geq U$.
$Q = \{q_0, \ldots, q_U\}$, and $\Gamma(q_i) = \overline{\mu_{\hat{\mathrm{S}}}(i)}$ for $i \leq U$.
$\delta(q_0, a) = 1$, $\delta(q_1, b) = 1$, $\delta(q_i, q_i, g) = 1$ and
$\delta(q_i, q_j, q_{main\{i+j,U\}}, f) = 1$ for $i, j \leq U$.

In modeling of Pattern 4.16, each tree with i nodes labeled b will lead to state q_i. Furthermore, $\hat{\mathrm{S}}$ is a fuzzy set that indicates significant numbers. So, $\Gamma(q_i) = \overline{\mu_{\hat{\mathrm{S}}}(i)}$ means that the fuzzy final states Γ interprets each state symbol q_i according to "significance of index i".

Let $P = \{p_1, \ldots, p_k\}$ is a set of properties and $AP = \{ap_1, \ldots, ap_m\}$ is the set of atomic propositions used in P. Now, we intend to define an FTA $M = (\Sigma, Q, \Gamma, \delta, \ell, \rho, \beta)$ that meets all of the properties in P, which means that $\beta \models p_1 \wedge \ldots \wedge p_k$. In this regard, we prove that the modeling of the combination of properties can be achieved by combining their models. Then, we provide an algorithm for combining the models $M_1, \ldots, M_m$, where $L(M_1) \models ap_1, \ldots, L(M_m) \models ap_m$, to achieve the model M.

Theorem 4.17. *Let $M_1 = (\Sigma, Q_1, \Gamma_1, \delta_1, \ell, \rho_1, \beta_1)$ and $M_2 = (\Sigma, Q_2, \Gamma_2, \delta_2, \ell, \rho_2, \beta_2)$ are two FTA that accept fuzzy tree languages $L(M_1)$ and $L(M_2)$ corresponding to atomic propositions ap_1 and ap_2, respectively. Then,*

I : There exists an FTA $M_3 = (\Sigma, Q_3, \Gamma_3, \delta_3, \ell, \rho_3, \beta_3)$ such that $L(M_3)$ is corresponding to $ap_1 \wedge ap_2$,
II : There exists an FTA $M_4 = (\Sigma, Q_4, \Gamma_4, \delta_4, \ell, \rho_4, \beta_4)$ such that $L(M_4)$ is corresponding to $ap_1 \vee ap_2$,

III : There exists an FTA $M_5 = (\Sigma, Q_5, \Gamma_5, \delta_5, \ell, \rho_5, \beta_5)$ *such that* $L(M_5)$ *is corresponding to* $\neg ap_1$.

Proof. According to the theorem, $L(M_3) = L(M_1) \cap L(M_2)$, $L(M_4) = L(M_1) \cup L(M_2)$ and $L(M_5) = \overline{L(M_1)}$. Since the class of fuzzy tree languages is closed under intersection, union and complement [4, 9], fuzzy tree languages $L(M_3)$, $L(M_4)$ and $L(M_5)$ are recognizable and FTA M_3, M_4 and M_5 are realizable. Now, we use tricks of modeling presented in Pattern 4.14 for constructing M_3. Also, we construct M_4 with the assumption that M_1 and M_2 are complete. Then, we construct M_5 with the assumption that M_1 is complete and deterministic and the membership grade of all its transition rules is 1.

I : $Q_3 = \Big\{q_{(i,j)} \mid q_i \in Q_1, q_j \in Q_2\Big\}, \forall q_{(i,j)} \in Q_3; \Gamma_3(q_{(i,j)}) = \Gamma_1(q_i) \wedge \Gamma_2(q_j)$ and $\forall \sigma \in \Sigma_n; \Big(\delta_1(q_1, \ldots, q_n, q_i, \sigma) = x_1, \delta_2(q_{1'}, \ldots, q_{n'}, q_j, \sigma) = x_2\Big) \Leftrightarrow \delta_3(q_{(1,1')}, \ldots, q_{(n,n')}, q_{(i,j)}, \sigma) = x_1 \wedge x_2$.
This settings implies that

$$\begin{aligned}
\beta_3(t) &= \bigvee_{q_{(i,j)} \in Q_3} \rho_3(t)(q_{(i,j)}) \wedge \Gamma_3(q_{(i,j)}) \\
&= \bigvee_{\substack{q_i \in Q_1 \\ q_j \in Q_2}} \Big(\rho_1(t)(q_i) \wedge \rho_2(t)(q_j)\Big) \wedge \Big(\Gamma_1(q_i) \wedge \Gamma_2(q_j)\Big) \\
&= \bigvee_{\substack{q_i \in Q_1 \\ q_j \in Q_2}} \Big(\rho_1(t)(q_i) \wedge \Gamma_1(q_i)\Big) \wedge \Big(\rho_2(t)(q_j) \wedge \Gamma_2(q_j)\Big) \\
&= \bigvee_{q_i \in Q_1} \rho_1(t)(q_i) \wedge \Gamma_1(q_i) \wedge \bigvee_{q_j \in Q_2} \rho_2(t)(q_j) \wedge \Gamma_2(q_j) \\
&= \beta_1(t) \wedge \beta_2(t).
\end{aligned}$$

So, $L(M_3) = L(M_1) \cap L(M_2)$.

II : $Q_4 = \Big\{q_{(i,j)} \mid q_i \in Q_1, q_j \in Q_2\Big\}, \forall q_{(i,j)} \in Q_4; \Gamma_4(q_{(i,j)}) = \Gamma_1(q_i) \vee \Gamma_2(q_j)$ and $\forall \sigma \in \Sigma_n; \Big(\delta_1(q_1, \ldots, q_n, q_i, \sigma) = x_1, \delta_2(q_{1'}, \ldots, q_{n'}, q_j, \sigma) = x_2\Big) \Leftrightarrow \delta_3(q_{(1,1')}, \ldots, q_{(n,n')}, q_{(i,j)}, \sigma) = x_1 \vee x_2$.
This settings implies that

$$\begin{aligned}
\beta_4(t) &= \bigvee_{q_{(i,j)} \in Q_4} \rho_4(t)(q_{(i,j)}) \wedge \Gamma_4(q_{(i,j)}) \\
&= \bigvee_{\substack{q_i \in Q_1 \\ q_j \in Q_2}} \Big(\rho_1(t)(q_i) \vee \rho_2(t)(q_j)\Big) \wedge \Big(\Gamma_1(q_i) \vee \Gamma_2(q_j)\Big) \\
&= \bigvee_{\substack{q_i \in Q_1 \\ q_j \in Q_2}} \Big(\rho_1(t)(q_i) \wedge \Gamma_1(q_i)\Big) \vee \Big(\rho_2(t)(q_j) \wedge \Gamma_2(q_j)\Big) \\
&= \bigvee_{q_i \in Q_1} \rho_1(t)(q_i) \wedge \Gamma_1(q_i) \vee \bigvee_{q_j \in Q_2} \rho_2(t)(q_j) \wedge \Gamma_2(q_j) \\
&= \beta_1(t) \vee \beta_2(t).
\end{aligned}$$

So, $L(M_4) = L(M_1) \cup L(M_2)$.

III : $Q_5 = Q_1$, $\Gamma_5 = \overline{\Gamma_1}$ and $\delta_5 = \delta_1$.
This settings implies that

$$\begin{aligned}\beta_5(t) &= \bigvee_{q\in Q_5} \rho_5(t)(q) \wedge \Gamma_5(q)\\ &= \rho_5(t)(q') \wedge \Gamma_5(q') = 1 \wedge \Gamma_5(q') = \Gamma_5(q')\\ &= \overline{\Gamma_1(q')} = \overline{1 \wedge \Gamma_1(q')} = \overline{\rho_1(t)(q') \wedge \Gamma_1(q')}\\ &= \overline{\bigvee_{q\in Q_1} \rho_1(t)(q) \wedge \Gamma_1(q)} = \overline{\beta_1(t)}.\end{aligned}$$

So, $L(M_5) = \overline{L(M_1)}$.

□

Corollary 4.18. *Assume that AP is a set of atomic propositions such that each $a \in AP$ is equivalent to a recognizable FTL. Then, any property which is a combination of the atomic propositions by $\vee$, $\wedge$, and $\neg$ can be modeled by an FTA.*

Proof. Assume that T is the binary expression tree of the modeling property, where its leaves are the atomic propositions, and its inner nodes are logical operations $\vee$, $\wedge$, and $\neg$. Algorithm 1 povides a systematic method for modeling T. In lines 2 and 3, each $a \in AP$ is modeled by FTA M_a, which means all leaves of T are modeled. Now, using an iterative loop (lines 4 to 6), the algorithm models each node whose children are modeled (Thm. 4.17) until all nodes of the tree are modeled. Clearly, the FTA corresponding to the root of T is the desired model of the property.

Algorithm 1: The modeling of a cpmplex property by an FTA.

input : P `// the input formula (a complex property)`
output: M `// the FTA model of the input formula`
1 $T \leftarrow expression\_tree(P)$ `// construct the expression tree`
2 **forall** $node \in T.leaves$ **do** `// modeling the atomic propositions`
3 $node.model \leftarrow modeling(node.property)$
4 **while** $T.root.model = \emptyset$ **do** `// bottom-up modeling of` T
5 $node \leftarrow T.get\_next()$ `// select a node that is not modeled but its children are modeled`
6 $node.model \leftarrow modeling(node.property)$ `// construct the model`

□

Corollary 4.19. *Assume that there exist a set of properties, each one equivalent to a recognizable FTL. Then, there exists an FTA satisfying all of these properties.*

Proof. Let $P = \{p_1, \ldots, p_n\}$ is the set of modeling properties, where $n \in \mathbb{N}$. Now, set $P_{all} = p_1 \wedge \ldots \wedge p_n$ and construct its FTA by Corollary 4.18. □

5. Conclusion

In this paper, we aimed to model a family of fuzzy tree languages in which some features of trees are described using linguistic variables. In this regard, we provided some heuristic tricks and techniques to employ fuzzy tree automata for modeling vaguely defined sets of regular trees on a ranked alphabet. Also, a general method of combining these models for constructing a synthetic fuzzy tree automaton is developed. This combination is based on the multiple indexing of states and the generalization of transition rules of fuzzy tree automata. It provides a systematic method for modeling a fuzzy tree language described by multiple properties. We also introduced a set of modeling patterns that encodes some crisp and fuzzy atomic propositions (simple properties) into fuzzy tree automata by defining the set of states, final states, and the set of transition rules of the desired automata. These modelings include some patterns such as symbol existence /counting, sub-structure detecting, the threshold on height and size, fuzzy numbers, as well as imprecision and vagueness handling, which are used in the linguistical description of fuzzy tree languages. In addition, we provided some examples that use multiple indexing of state symbols for combining the models, and then we proved that if two properties are satisfiable by some fuzzy tree languages, their combination with logical modalities such as $\vee$, $\wedge$, and $\neg$ can be modeled by some fuzzy tree automata (Thm. 4.17). As a result of the theorem, every complex property or even a set of properties can be organized into a binary expression tree to construct a fuzzy tree automaton model recognizing the corresponding fuzzy tree language.

References

[1] R. Bača, M. Krátkỳ, I. Holubová, M. Nečaskỳ, T. Skopal, M. Svoboda and S. Sakr, Structural XML query processing. *ACM Comput. Surv.* **50** (2017) 1–41.

[2] T. Ballard, H. Palada, M. Griffin and A. Neal, An integrated approach to testing dynamic, multilevel theory: using computational models to connect theory, model, and data. *Org. Res. Methods* **24** (2021) 251–284.

[3] S. Borgwardt and R.P. naloza, Reasoning in fuzzy description logics using automata. *Fuzzy Sets Syst.* **298** (2016) 22–43.

[4] S. Bozapalidis and O.L. Bozapalidoy, Fuzzy tree language recognizability. *Fuzzy Sets Syst.* **161** (2010) 716–734.

[5] S. Chehida, A. Baouya, S. Bensalem and M. Bozga, Learning and analysis of sensors behavior in IoT systems using statistical model checking. *Softw. Quality J.* (2021) 1–22. Available from: https://doi.org/10.1007/s11219-021-09559-w

[6] K.C. Clarke, Mathematical Foundations of Cellular Automata and Complexity Theory, in The Mathematics of Urban Morphology. Springer (2019) 163–170.

[7] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, C. Loding, S. Tison and M. Tommasi, Tree automata: techniques and applications (2007). Preprint https://hal.inria.fr/hal-03367725

[8] Y. Du and P. Zhu, Fuzzy approximations of fuzzy relational structures. *Int. J. Approx. Reas.* **98** (2018) 1–10.

[9] Z. Esik and L. Guangwu, Fuzzy tree automata. *Fuzzy Sets Syst.* **158** (2007) 1450–1460.

[10] M.K. Fallah, S. Moghari, E. Nazemi and M.M. Zahedi, Fuzzy ontology based document feature vector modification using fuzzy tree transducer, in *Proceedings of the 2008 IEEE International Conference on Signal Image Technology and Internet Based Systems* (2008) 38–44.

[11] A.-J. Fougères and E. Ostrosi, Fuzzy engineering design semantics elaboration and application. *Soft Comput. Lett.* **3** (2021) 100025.

[12] H. Frenkel, O. Grumberg and S. Sheinvald, An automata-theoretic approach to model-checking systems and specifications over infinite data domains. *J. Autom. Reas.* **63** (2019) 1077–1101.

[13] M. Ghorani, S. Garhwal and S. Moghari, Lattice-valued tree pushdown automata: pumping lemma and closure properties. *Int. J. Approx. Reas.* **142** (2022) 307–323.

[14] M. Ghorani and S. Moghari, Decidability of the minimization of fuzzy tree automata with membership values in complete lattices. *J. Appl. Math. Comput.* **68** (2022) 461–478.

[15] E. González-Caballero, R.A. Espín-Andrade, W. Pedrycz, L. Martínez and L.A. Guerrero-Ramos, Continuous linguistic variables and their applications to data mining and time series prediction. *Int. J. Fuzzy Syst.* (2021) 1–22.

[16] P. Grzegorzewski, Metrics and orders in space of fuzzy numbers. *Fuzzy Sets Syst.* **97** (1998) 83–94.

[17] M. Hachicha and J. Darmont, A survey of XML tree patterns. *IEEE Trans. Knowl. Data Eng.* **25** (2011) 29–46.

[18] J. He, X. Li, Y. Yao, Y. Hong and Z. Jinbao, Mining transition rules of cellular automata for simulating urban expansion by using the deep learning techniques. *Int. J. Geogr. Inf. Sci.* **32** (2018) 2076–2097.

[19] M. He and S. Kazi, Data structures for categorical path counting queries, in *32nd Annual Symposium on Combinatorial Pattern Matching (CPM 2021), Schloss Dagstuhl-Leibniz-Zentrum für Informatik* (2021).

[20] J.E. Hopcroft, Introduction to automata theory, languages, and computation. Pearson Education India (2008).

[21] F. Howar and B. Steffen, Active automata learning in practice, in Machine Learning for Dynamic Software Analysis: Potentials and Limits. Springer (2018) 123–148.

[22] X. Ji, L. Wang, H. Xue and Y. Gao, Decision-making method of qualitative and quantitative comprehensive evaluation of talents based on probability hesitation fuzzy language. *Math. Probl. Eng.* **2021** (2021) 8903427.

[23] K. Johannisson, Disambiguating implicit constructions in OCL, in Workshop on OCL and Model Driven Engineering at UML2004, Lisbon (2004).
[24] A. Kundu and E. Bertino, Structural signatures for tree data structures. *Proc. VLDB Endow.* **1** (2008) 138–150.
[25] I. Lamrani, A. Banerjee and S.K. Gupta, Hymn: Mining linear hybrid automata from input output traces of cyber-physical systems, in *2018 IEEE Industrial Cyber-Physical Systems (ICPS), IEEE* (2018) 264–269.
[26] X. Li and A. Gar-On Yeh, Data mining of cellular automata's transition rules. *Int. J. Geogr. Inf. Sci.* **18** (2004) 723–744.
[27] Y. Li, Approximation and robustness of fuzzy finite automata. *Int. J. Approx. Reas.* **47** (2008) 247–257.
[28] D.L. Ly and H. Lipson, Learning symbolic representations of hybrid dynamical systems. *J. Mach. Learn. Res.* **13** (2012) 3585–3618.
[29] R. Medhat, S. Ramesh, B. Bonakdarpour and S. Fischmeister, A framework for mining hybrid automata from input/output traces, in *Proceedings of the 12th International Conference on Embedded Software*. IEEE Press (2015) 177–186.
[30] S. Moghari and M. Ghorani, A symbiosis between cellular automata and dynamic weighted multigraph with application on virus spread modeling. *Chaos Solitons Fract.* **155** (2022) 111660.
[31] S. Moghari and M. Zahedi (1711-4091), Multidimensional fuzzy finite tree automata. *Iran. J. Fuzzy Syst.* **16** (2019) 155–167.
[32] S. Moghari and M.M. Zahedi, Similarity-based minimization of fuzzy tree automata. *J. Appl. Math. Comput.* **50** (2016) 417–436.
[33] S. Moghari, M.M. Zahedi and R. Ameri, New direction in fuzzy tree automata. *Iranian J. Fuzzy Syst.* **8** (2011) 59–68.
[34] S. Mohammed, A.F. Barradah and E.-S.M. El-Alfy, Selectivity estimation of extended XML query tree patterns based on prime number labeling and synopsis modeling. *Simul. Model. Practice Theory* **64** (2016) 30–42.
[35] S. Mohammed, E.-S.M. El-Alfy and A.F. Barradah, Improved selectivity estimator for XML queries based on structural synopsis. *World Wide Web* **18** (2015) 1123–1144.
[36] J. Mordeson and D.S. Malik, Fuzzy automata and languages: theory and applications. Chapman & Hall, London (2002).
[37] L.D. Nguyen and D.Q. Tran, Measurement of fuzzy membership functions in construction risk assessment. *J. Constr. Eng. Manag.* **147** (2021) 04021005.
[38] B.E. Reddy, R.O. Reddy and E.K. Reddy, Pattern analysis and texture classification using finite state automata scheme. *Int. J. Adv. Intell. Parad.* **14** (2019) 30–45.
[39] M.S. Roodposhti, J. Aryal and B.A. Bryan, A novel algorithm for calculating transition potential in cellular automata models of land-use/cover change. *Environ. Model. Softw.* **112** (2019) 70–81.
[40] M.G. Soto, T.A. Henzinger, C. Schilling and L. Zeleznik, Membership-based synthesis of linear hybrid automata, in International Conference on Computer Aided Verification. Springer (2019) 297–314.
[41] J.L. Verdegay, Vol. 377 of Uncertainty Management with Fuzzy and Rough Sets. Springer (2019).
[42] L. Wang, Y. Wang, D. Cai, D. Zhang and X. Liu, Translating a math word problem to an expression tree. Preprint arXiv:1811.05632 (2018).
[43] L.-G. Wang, T.T.-Y. Lam, S. Xu, Z. Dai, L. Zhou, T. Feng, P. Guo, C.W. Dunn, B.R. Jones, T. Bradley *et al.*, Treeio: an R package for phylogenetic tree input and output with richly annotated and associated data. *Mol. Biol. Evol.* **37** (2020) 599–603.
[44] X. Wu and D. Theodoratos, Template-based bitmap view selection for optimizing queries over tree data. *Int. J. Cooperative Inf. Syst.* **25** (2016) 1650005.
[45] C. Yang and Y. Li, Approximate bisimulations and state reduction of fuzzy automata under fuzzy similarity measures. *Fuzzy Sets Syst.* **391** (2020) 72–95.
[46] H. Ying and F. Lin, Online self-learning fuzzy discrete event systems. *IEEE Trans. Fuzzy Syst.* **28** (2020) 2185–2194.
[47] M. Yulduz, Lexico-grammatical parts of speech expressing the indefiniteness of the subject. *JournalNX* **7** (2021) 323–327.
[48] L.A. Zadeh, Fuzzy sets. *Inf. Control* **8** (1965) 338–353.
[49] L.A. Zadeh, The concept of a linguistic variable and its application to approximate reasoning-I. *Inf. Sci.* **8** (1975) 199–249.
[50] L.A. Zadeh, The concept of a linguistic variable and its application to approximate reasoning-II. *Inf. Sci.* **8** (1975) 301–357.
[51] H.J. Zimmermann, Fuzzy set theory and its applications, 3rd edn. Springer Science & Business Media (2011).