TREE AUTOMATA BASED ON COMPLETE RESIDUATED LATTICE-VALUED LOGIC: REDUCTION ALGORITHM AND DECISION PROBLEMS

M. GHORANI

Abstract. In this paper, at first we define the concepts of response function and accessible states of a complete residuated lattice-valued (for simplicity we write $\mathcal{L}$-valued) tree automaton with a threshold c. Then, related to these concepts, we prove some lemmas and theorems that are applied in considering some decision problems such as finiteness-value and emptiness-value of recognizable tree languages. Moreover, we propose a reduction algorithm for $\mathcal{L}$-valued tree automata with a threshold c. The goal of reducing an $\mathcal{L}$-valued tree automaton is to obtain an $\mathcal{L}$-valued tree automaton with reduced number of states all of which are accessible, in addition it recognizes the same language as the first one given. We compare our algorithm with some other algorithms in the literature. Finally, utilizing the obtained results, we consider some fundamental decision problems for $\mathcal{L}$-valued tree automata including the membership-value, the emptiness-value, the finiteness-value, the intersection-value and the equivalence-value problems.

1. Introduction

Tree automata deal with finite ordered ranked trees which correspond to finite terms over a ranked alphabet. The theory of tree automata is an extension of word automata theory where words are viewed as unary terms. Tree automata were initially introduced by Doner [14, 15] and Thatcher and Wright [47]. Researchers have applied tree automata in considering decidability results in logic [13], abstract interpretation using set constraints [19] and program verification [25]. Tree automata based on fuzzy logic have been considered by some researches. For instance, Ésik and Liu [18] obtained a Kleene theorem for fuzzy tree languages. Moreover, Moghari et al. [37] considered the behaviour of fuzzy tree automata. Also, application of fuzzy tree automata in the fuzzy computation tree logic model checking was studied in [31, 32, 39, 40]. Other various issues and applications of fuzzy tree automata have been investigated by researchers (see [6, 36, 38]).

Residuated lattices are important algebras and are closely related to various important logics such as MV-algebras, product algebras, and Gödel algebras [5, 41].

Received: June 2016; Revised: January 2017; Accepted: September 2017

Key words and phrases: Tree automata, Lattice-valued logic, Reduction algorithm, Decision problem, Time complexity.

This paper is the extended version of presented paper in "International Conference on Mathematics of Fuzziness (ICMF)" which was held during 27-29 April 2016 at Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan, Iran.

Automata theory based on complete residuated lattice-valued logic was proposed by Qiu [42, 43]. In recent years, he and several other authors have studied lattice-valued automata (see [2, 27, 30, 33, 34, 44, 48, 49, 50]). Furthermore, Ghorani and Zahedi [21, 22, 23] and Ghorani et al. [24] investigated tree automata over complete residuated lattices.

Reduction of a tree automaton means obtaining a smaller tree automaton with the same language. The reason for performing reduction is that the smaller reduced tree automaton is more efficient to handle in a subsequent computation. Thus, there is an algorithmic tradeoff between the effort for reduction and the complexity of the problem which is going to be considered for this tree automaton. Efficient reduction algorithms have been presented both for word automata [26, 45] and tree automata [1, 3, 4, 12]. For instance, Abdulla et al. [1] combined various upward and downward bisimulation and simulation relations to obtain a reduced tree automaton. Also, Almeida [3] obtained a polynomial time reduction algorithm by using a technique that adds new transitions to a tree automaton. Moreover, state reduction of fuzzy automata has been studied by many authors (refer to [8, 38]). In [9, 10, 46] the authors have studied fuzzy automata over a complete residuated lattice, and showed that by using fuzzy equivalences and fuzzy quasi-orders instead of crisp ones, one can attain better reductions. In [21, 35, 36], the authors presented reduction algorithms for complete and deterministic lattice-valued tree automata.

Different fuzzy decision problems for (tree) automata have been investigated in numerous papers. For instance, Kupferman and Lustig [28] studied fundamental decision problems such as emptiness-value, finiteness-value, and implication-value problems for lattice automata and showed that the mentioned problems have the same complexities as the corresponding ones in the Boolean setting. The computation of the behaviour of weighted tree automata over trees, as a decision problem, was investigated in [16, 17, 29].

Due to various applications of tree automata, it is very important to reach their accessible states. As well, the boundaries of some decidability problems are dependent on the complexity of reduction methods. Therefore, in this paper, at first, we define the concepts of response function and accessible states of an $\mathcal{L}$-valued tree automaton with a threshold c, then we obtain some lemmas that are useful for solving some decision problems. Subsequently, we provide an efficient reduction algorithm for an $\mathcal{L}$-valued tree automaton with a threshold c. Time complexity of the given algorithm is discussed and the proposed algorithm is compared with some existing reduction algorithms in the literature [12, 21, 35, 36]. Furthermore, by applying the obtained results we study some decision problems containing the membership-value, the emptiness-value, the finiteness-value, the intersection-value and the equivalence-value problems.

The paper is arranged in six sections as follows: Section 2, contains some preliminaries and basic definitions that are needed for remaining parts of the paper. In Section 3, we study response function and accessible states of $\mathcal{L}$-valued tree automata, and obtain some useful results that are used in decision problems. In Section 4, we construct a reduction algorithm for a given $\mathcal{L}$-valued tree automaton and investigate its time complexity. In Section 5, we consider some decision

problems and investigate their time complexities. Finally, Conclusions are given in Section 6.

2. Preliminaries and Basic Definitions

In this section we present some terminologies and definitions on tree automata theory and lattice-valued logic that will be used throughout the paper. For more details, we refer to [5, 12, 20, 21, 44].

Definition 2.1. [41] A complete residuated lattice is a 5-tuple $l =< \mathcal{L}, +, \cdot, \odot, \rho >$ where:

(i) $< \mathcal{L}, +, \cdot >$ is a complete lattice with the least and the greatest elements 0 and 1, respectively;

(ii) $\odot$ and ρ are two binary operations on $\mathcal{L}$ such that $\odot$ is isotone, $< \mathcal{L}, \odot, 1 >$ is a commutative monoid, and ρ is antitone in the first and isotone in the second variable; that is, for any $a_1, a_2, b \in \mathcal{L}$ if $a_1 \leq a_2$ then $a_1 \odot b \leq a_2 \odot b$, $b \odot a_1 \leq b \odot a_2$, $a_2 \rho b \leq a_1 \rho b$ and $b \rho a_1 \leq b \rho a_2$;

(iii) for all $a, b, c \in \mathcal{L}$, $a \odot b \leq c$ if and only if $a \leq b \rho c$.

Remark 2.2. In this paper, by an $\mathcal{L}$-valued logic we mean the complete residuated lattice-valued logic; that is, the set of truth values is $\mathcal{L}$, which possesses nullary connective a ($a \in \mathcal{L}$), an additional binary connective & as well as usual connectives $\vee, \wedge$ and implication $\rightarrow$. Moreover, in $\mathcal{L}$-valued logic a formula φ is valid if and only if $[\varphi] = 1$ for any interpretation, where $[\varphi]$ stands for the truth value of φ (see [21, 41, 44]). Now, we present the truth valuation rules of the predicate logical and set theoretical formulas as follows:

(i)
$$[a] = a(a \in \mathcal{L}),\ [\varphi \vee \psi] = [\varphi] + [\psi],\ [\varphi \wedge \psi] = [\varphi] \cdot [\psi],$$
$$[\varphi \rightarrow \psi] = [\varphi] \rho [\psi],\ [\varphi \& \psi] = [\varphi] \odot [\psi].$$

(ii) If X is the universe (the set which contains all objects), $\mathcal{L}^X$ denotes the class of $\mathcal{L}$-valued subsets of X over $\mathcal{L}$, where an $\mathcal{L}$-valued subset of X means a mapping from X to $\mathcal{L}$. Now, let $A \in \mathcal{L}^X$. Then, $[x \in A] = A(x)$.

In this paper, we assume that the complete residuated lattice is also a chain, i.e. all pairs of elements are comparable.

Example 2.3. [21] Let $\mathcal{L} = [0, 1] \subseteq \mathbb{R}$. Then $l =< \mathcal{L}, \vee, \wedge, \odot, \rho >$ is a complete residuated lattice, where $a \odot b = \max(0, a + b - 1), a \rho b = \min(1, 1 - a + b)$ for any $a, b \in [0, 1]$, and $\vee$ and $\wedge$ are the symbols of the truth-valued lattice, representing max and min, respectively.

Example 2.4. For all $a, b \in [0, 1] \subseteq \mathbb{R}$, let $a \odot b = min\{a, b\} = a \wedge b$ and

$$a \rho b = \begin{cases} 1, & if\ a \leq b, \\ b, & otherwise. \end{cases}$$

Then $l =< [0, 1], \vee, \wedge, \odot, \rho >$ is a complete residuated lattice, where $\vee$ and $\wedge$ are max and min, respectively.

In the following, we provide an example that is not a complete residuated lattice. Here, the conditions (i) and (ii) of Definition 2.1 hold but the condition (iii) does not hold.

Example 2.5. [7] Let $\mathcal{L} = [-4, 0] \subseteq \mathbb{R}$ with natural ordering and $a, b \in \mathcal{L}$. Define

$$a \odot b = \begin{cases} a + b - ab, & if\ a, b \in Q, \\ a + b, & if\ a \text{ or } b \in Q^c, \end{cases}$$

and

$$a\rho b = \begin{cases} min\{0, \frac{b-a}{1-a}\}, & if\ a, b \in Q, \\ min\{0, b - a\}, & if\ a \text{ or } b \in Q^c, \end{cases}$$

where, Q and Q^c show the set of rational and irrational numbers, respectively. It is obvious that $< \mathcal{L}, \vee, \wedge >$ is a complete lattice ($\vee$ and $\wedge$ are max and min, respectively). Now, we show that condition (ii) of Definition 2.1 holds. It is obvious that $< \mathcal{L}, \odot, 0 >$ is a commutative monoid. To show that for any $a_1, a_2, b \in \mathcal{L}$ if $a_1 \leq a_2$ then $a_1 \odot b \leq a_2 \odot b$, $b \odot a_1 \leq b \odot a_2$, $a_2\rho b \leq a_1\rho b$ and $b\rho a_1 \leq b\rho a_2$, we must consider the following eight cases:

(1) $a_1, a_2, b \in Q$,
(2) $a_1 \in Q$ and $a_2, b \in Q^c$,
(3) $a_1, a_2 \in Q$ and $b \in Q^c$,
(4) $a_1, b \in Q$ and $a_2 \in Q^c$,
(5) $a_1 \in Q^c$ and $a_2, b \in Q$,
(6) $a_1, a_2 \in Q^c$ and $b \in Q$,
(7) $a_1, b \in Q^c$ and $a_2 \in Q$,
(8) $a_1, a_2, b \in Q^c$.

For all of the above cases we can consider the following three items:

(I1) $b \leq a_1 \leq a_2$,
(I2) $a_1 \leq b \leq a_2$,
(I3) $a_1 \leq a_2 \leq b$.

We consider case (1)-(I1). Let $a_1, a_2, b \in Q$, and $b \leq a_1 \leq a_2$. At first we demonstrate that $b\rho a_1 \leq b\rho a_2$. In fact, we show that

$$min\{0, \frac{a_1 - b}{1 - b}\} \leq min\{0, \frac{a_2 - b}{1 - b}\}. \tag{1}$$

Since $\frac{a_1-b}{1-b} \geq 0$ and $\frac{a_2-b}{1-b} \geq 0$, we have $min\{0, \frac{a_1-b}{1-b}\} = min\{0, \frac{a_2-b}{1-b}\} = 0$. Thus, relation (1) holds.
Now, we show that $a_2\rho b \leq a_1\rho b$, that is:

$$min\{0, \frac{b - a_2}{1 - a_2}\} \leq min\{0, \frac{b - a_1}{1 - a_1}\}. \tag{2}$$

If $b = a_1 = a_2$, then $min\{0, \frac{b-a_2}{1-a_2}\} = min\{0, \frac{b-a_1}{1-a_1}\} = 0$. Therefore relation (2) holds.
If $b = a_1 < a_2$, then $min\{0, \frac{b-a_1}{1-a_1}\} = 0$ and $min\{0, \frac{b-a_2}{1-a_2}\} < 0$. Therefore relation (2) holds.

If $b < a_1 = a_2$, then $min\{0, \frac{b-a_1}{1-a_1}\} = min\{0, \frac{b-a_2}{1-a_2}\}$. Therefore relation (2) holds.
If $b < a_1 < a_2$, then it is enough to show that $\frac{b-a_2}{1-a_2} < \frac{b-a_1}{1-a_1}$. Since $1-a_1 > 1-a_2 > 0$, $\frac{1}{1-a_2} > \frac{1}{1-a_1} > 0$ and $b-a_2 < b-a_1 < 0$, we conclude that $\frac{b-a_2}{1-a_2} < \frac{b-a_1}{1-a_1}$. Therefore relation (2) holds.
Now we show that $a_1 \odot b \leq a_2 \odot b$, i.e. $a_1 + b - a_1 b \leq a_2 + b - a_2 b$. Since $a_1 \leq a_2$ and $b \leq 0$, we have $-a_1 b \leq -a_2 b$. Thus $a_1 - a_1 b \leq a_2 - a_2 b$, that is $a_1 \odot b \leq a_2 \odot b$. Clearly, $b \odot a_1 \leq b \odot a_2$. By a similar method we can consider the other cases. Therefore, condition (ii) holds. However, condition (iii) does not hold because $-1 \odot -1 = -3 \leq -\sqrt{5}$, but $-1 > -1\rho - \sqrt{5} = -\sqrt{5}+1$. Hence, $l =< \mathcal{L}, \vee, \wedge, \odot, \rho >$ is not a complete residuated lattice.

In the following, we give an example that has conditions (i) and (iii) but does not have condition (ii).

Example 2.6. Suppose that $l =< \mathcal{L}, \vee, \wedge, \odot, \rho >$, where $\mathcal{L} = \{0, a, b, c, 1\}$ with $0 < a < b, c < 1$, and b and c are incomparable, $\odot = \wedge$ and ρ is defined as Table 1. It is easy to check that conditions (i) and (iii) hold. However condition (ii) does not hold, becuase $a \leq c$ but $b\rho a > b\rho c$.

ρ	0	a	b	c	1
0	1	1	1	1	1
a	0	1	1	1	1
b	0	1	1	c	1
c	0	b	b	1	1
1	0	a	b	c	1

TABLE 1. The Operation ρ in Example 2.6

Example 2.7. Let Σ be an alphabet with more than one element, Σ^* be the set of all words over Σ and $\mathcal{L} = 2^{\Sigma^*}$ (the power set of Σ^*) be the set of all formal languages over Σ. Then, $< \mathcal{L}, \cup, \cap >$ is not a complete lattice. Let $l =< \mathcal{L}, \cup, \cap, \odot, \rightarrow, \leftarrow >$ where for all $M, N \in \mathcal{L}$, $M \odot N = MN = \cup_{\alpha \in M, \beta \in N} \alpha\beta$, $M \rightarrow N = \{w \mid w \in \Sigma^*, Mw \subseteq N\}$, $M \leftarrow N = \{w \mid w \in \Sigma^*, wM \subseteq N\}$ and the monoid unit is the language $\{\varepsilon\}$ consisting of just the empty string ε. It can be seen that (iii) holds but (i) and (ii) do not. Now, assume that Σ has just one element. Then $< \mathcal{L}, \odot, \varepsilon >$ is commutative. Thus (ii) and (iii) hold but (i) does not. Moreover, if we assume that Σ is empty then the language on Σ is finite. Thus (i),(ii) and (iii) hold.

Remark 2.8. In the above example since the monoid is not commutative, we consider both residuals $M \rightarrow N$ and $M \leftarrow N$. Note that, if monoid is commutative then both residuals are the same operation, namely $M\rho N$ (refer to [11]).

In Example 2.7 we see that if Σ has just one element then (ii) and (iii) hold but (i) does not. Therefore, Examples 2.4, 2.6 and 2.7 show that the three conditions (i),(ii) and (iii) in Definition 2.1 are independent.

Example 2.9. Let $\mathcal{L} = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$, where $\emptyset \subseteq \{a\}, \{b\} \subseteq \{a, b\}$ and ρ and $\odot$ are as Table 2. Then, $l =< \mathcal{L}, \cup, \cap, \odot, \rho >$ is a complete residuated lattice, which is not chain.

$\odot$	$\emptyset$	{a}	{b}	{a,b}
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
{a}	$\emptyset$	{a}	$\emptyset$	{a}
{b}	$\emptyset$	$\emptyset$	{b}	{b}
{a,b}	$\emptyset$	{a}	{b}	{a,b}

ρ	$\emptyset$	{a}	{b}	{a,b}
$\emptyset$	{a,b}	{a,b}	{a,b}	{a,b}
{a}	{b}	{a,b}	{b}	{a,b}
{b}	$\emptyset$	{a}	{a,b}	{a,b}
{a,b}	$\emptyset$	{a}	{b}	{a,b}

TABLE 2. The Operations $\odot$ and ρ in Example 2.9

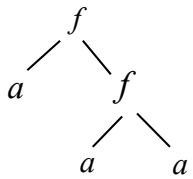
Definition 2.10. [12] A ranked alphabet is a couple $(F, Arity)$ where F is a finite set and $Arity$ is a mapping from F into $\mathbb{W}$ (the set of non-negative integers). The arity of a symbol $f \in F$ is $Arity(f)$. The set of symbols of arity n is denoted by F_n. Here, we use parenthesis and commas for a short declaration of symbols with arity. For instance, $f(,)$ is a short declaration for a binary symbol f.

Definition 2.11. [12] The set $\mathcal{T}(F, X)$ of terms, over the ranked alphabet F and the set of variables X, is the smallest set defined by:
$-$ $F_0 \subseteq \mathcal{T}(F, X)$,
$-$ $X \subseteq \mathcal{T}(F, X)$ and
$-$ if $p \geq 1$, $f \in F_p$ and $t_1, \cdots, t_p \subseteq \mathcal{T}(F, X)$, then $f(t_1, \cdots, t_p) \in \mathcal{T}(F, X)$.
If $X = \emptyset$, we write $\mathcal{T}(F)$ rather than $\mathcal{T}(F, X)$. Terms in $\mathcal{T}(F)$ are called ground terms. A $t \in \mathcal{T}(F, X)$ is linear if each variable occurs at most once in t.

Example 2.12. Let $F = \{a, f(,)\}$ and $X = \{x\}$. Hear f is a binary symbol and a is a constant. We have

$$\mathcal{T}(F, X) = \{a, x, f(a, a), f(a, x), f(x, a), f(x, x), f(x, f(x, a)), f(a, f(a, a)), \cdots\}.$$

The terms $f(a, x), f(x, a)$ and $f(a, f(a, x))$ are linear; the terms $f(x, x)$ and $f(x, f(x, a))$ are nonlinear; the terms $f(a, a)$ and $f(a, f(a, a))$ are ground terms. Terms can be represented in a graphical way. For instance, the term $f(a, f(a, a))$ is represented by:

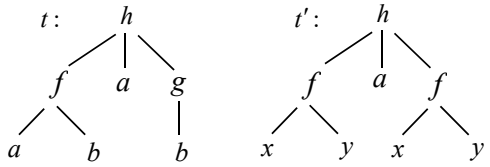


Definition 2.13. [12] Let $\mathbb{W}^*$ be the set of finite strings over $\mathbb{W}$, and let ε be the empty string. A term $t \in \mathcal{T}(F, X)$ can be defined as a partial function $t : \mathbb{W}^* \to F \cup X$ with domain $pos(t)$, satisfying the following properties:
(i) $\mathcal{T}(\varepsilon) = Head(t)$, where $Head(t)$ is the root symbol of t,
(ii) $pos(t)$ is non-empty and prefix-closed,
(iii) $\forall p \in pos(t)$, if $\mathcal{T}(p) \in F_n$, $n \geq 1$, then $\{j | pj \in pos(t)\} = \{0, 1, \cdots, n-1\}$,
(iv) $\forall p \in pos(t)$, if $\mathcal{T}(p) \in X \cup F_0$, then $\{j | pj \in pos(t)\} = \emptyset$.
Each element in $pos(t)$ is called a position.

Definition 2.14. [12] A subterm $t|_p$ of a term $t \in \mathcal{T}(F, X)$ at position p is defined as follows:
- $pos(t|_p) = \{j |\ pj \in pos(t)\}$,
- $\forall q \in pos(t|_p),\ t|_p(q) = \mathcal{T}(pq)$.

Definition 2.15. [12] The height of a term t, denoted by $Height(t)$, is inductively defined by:
– $Height(t) = 0$ if $t \in X$,
– $Height(t) = 1$ if $t \in F_0$,
– $Height(t) = 1 + \max\{Height(t_i) |\ i \in \{1, \cdots, n\}\}$ if $Head(t) \in F_n$.

Example 2.16. Let $F = \{a, b, g(), f(,), h(,,)\}$ and $X = \{x, y\}$. Consider the terms t and t', as follows:



The root symbol of t is h; the root symbol of t' is h; the set of positions of t is $\{\varepsilon, 1, 2, 3, 11, 12, 31\}$; the set of positions of t' is $\{\varepsilon, 1, 2, 3, 11, 12, 31, 32\}$; $Height(t) = 3$; $Height(t') = 2$; $t|_3 = g(b)$; $t'|_3 = f(x, y)$.

Definition 2.17. [12] Let $X_n = \{x_1, \cdots, x_n\}$ be a set of n variables. A linear term $C \in \mathcal{T}(F, X_n)$ is called a context and the expression $C[t_1, \cdots, t_n]$ for $t_1, \cdots, t_n \in \mathcal{T}(F)$ denotes the term in $\mathcal{T}(F)$ obtained from C by replacing variable x_i by t_i for each $1 \leq i \leq n$, that is $C[t_1, \cdots, t_n] = C\{x_1 \leftarrow t_1, \cdots, x_n \leftarrow t_n\}$. $\mathcal{C}^n(F)$ denotes the set of contexts over $(x_1, \cdots, x_n)$ and $\mathcal{C}(F)$ denotes the set of contexts containing a single variable. A context is trivial if it is reduced to a variable.

Definition 2.18. [21] A tuple $A = (Q, F, Q_f, \delta)$ is called an $\mathcal{L}$-valued tree automaton, where Q is a finite set of states, F is a ranked alphabet, Q_f is an $\mathcal{L}$-valued set of final states, and δ is an $\mathcal{L}$-valued subset of $Q^n \times Q \times F_n$, i.e. a mapping from $Q^n \times Q \times F_n$ to $\mathcal{L}$, for each $n \geq 0$.

Remark 2.19. The family of $\mathcal{L}$-valued subsets $\delta = \cup_{i=1}^{m} \delta_i$, is called transition, where $|\delta| = m$ is the cardinality of δ.

3. Response Function and Accessible States

In this section, we define the concepts of response function and accessible states of an $\mathcal{L}$-valued tree automaton with a threshold c. Also, we provide some results that are used in Section 5 for solving some decision problems.
From now on, we assume that $A = (Q, F, Q_f, \delta)$ is an $\mathcal{L}$-valued tree automaton and $c \in \mathcal{L} \setminus \{1\}$.

Definition 3.1. The response function $r_A : \mathcal{T}(F) \times Q \to \mathcal{L}$ of A, by induction on $t \in \mathcal{T}(F)$, is defined as follows:

(i) If $t = \sigma \in F_0$, then $[r_A(t, q)] = \delta(\varepsilon, q, t),\ \forall q \in Q$.
(ii) If $t = \sigma(t_1, \cdots, t_n),\ \forall t_1, \cdots, t_n \in \mathcal{T}(F)$ and $\sigma \in F_n$, then

$$[r_A(t, q)] = \sum_{(q_1, \cdots, q_n) \in Q^n} (\delta((q_1, \cdots, q_n), q, \sigma). \prod_{i=1}^{n} [r_A(t_i, q_i)]).$$

For simplicity, we use $\delta(q, t)$ instead of $\delta(\varepsilon, q, t)$.

In fact, the response function r_A shows the level of access to $q \in Q$ by $t \in \mathcal{T}(F)$. The behaviour of A with a threshold c is defined as follows:

$$B(A,c) = \{t \in \mathcal{T}(F) | \sum_{q \in Q} ([r_A(t,q)].Q_f(q)) > c\}.$$

Also, the term language $L^c(A)$, recognized by A with a threshold c, is defined as:

$$L^c(A) = \{t \in \mathcal{T}(F) | \ t \in B(A,c)\}.$$

The set $\bar{L}$ is recognizable with a threshold c if there exists an $\mathcal{L}$-valued tree automaton A such that $\bar{L} = L^c(A)$. The size of $\mathcal{L}$-valued tree automaton A, denoted by $\|A\|$, is defined as follows:

$$\|A\| = |Q| + \sum_{((q_1,\cdots,q_n),q,f)\in\delta} (Arity(f) + 3).$$

Definition 3.2. Let A be an $\mathcal{L}$-valued tree automaton over the ranked alphabet F, and $t \in \mathcal{T}(F)$. An $\mathcal{L}$-valued run r of A on t with a threshold c is a mapping $r : pos(t) \times Q \to \mathcal{L}$ compatible with δ, i.e., $\forall p, p_i \in pos(t)$, if $t(p) = \sigma \in F_n, [r(p,q)] > c$ and $[r(p_i,q_i)] > c, \forall i \in \{1,\cdots,n\}$, then $\delta((q_1,\cdots,q_n),\sigma,q) > c$.

Definition 3.3. Let $q \in Q$. Then, q is called an accessible state of A with a threshold c if there exists $t \in \mathcal{T}(F)$ such that $[r_A(t,q)] > c$. Also, the state $q \in Q$ is called an accessible state of A if for all $c \in \mathcal{L} \setminus \{1\}$, q is an accessible state of A with threshold c.
The set of all accessible states of A with a threshold c, is denoted by,

$$Q_c = \{q \in Q | \ \exists t \in \mathcal{T}(F), \ [r_A(t,q)] > c\}.$$

The set of all accessible states is denoted by $\bar{Q}$. Hence, $\bar{Q} = \cap_{c\in\mathcal{L}\setminus\{1\}} Q_c$.

Now, we provide an example to illustrate the accessible states of an $\mathcal{L}$-valued tree automaton A with a threshold c.

Example 3.4. Let $\mathcal{L} = [0,1] \subseteq \mathbb{R}$, $c = 0.2$ and $A = (Q, F, Q_f, \delta)$, where $Q = \{q_0,q_1,q_2,q_3\}$, $F = \{a, b, g(), \ f(,)\}$, $Q_f(q_0) = 0.2$, $Q_f(q_1) = 0.4$, $Q_f(q_2) = 0.7$, $Q_f(q_3) = 0.5$ and δ is as follows:

$$\delta_1(q_0,a) = 0.4, \ \delta_2(q_1,b) = 1,$$
$$\delta_3(q_0,q_2,g) = 0.2, \ \delta_4(q_0,q_1,g) = 0.3,$$
$$\delta_5((q_2,q_1),q_3,f) = 0.2, \ \delta_6((q_1,q_1),q_3,f) = 0.3.$$

Some of the response functions of $\mathcal{L}$-valued tree automaton A are as follows:

$$[r_A(a,q_0)] = 0.4, \ [r_A(b,q_1)] = 1,$$
$$[r_A(g(a),q_2)] = [r_A(a,q_0)] \cdot \delta_3(q_0,q_2,g) = 0.2,$$
$$[r_A(g(a),q_1)] = [r_A(a,q_0)] \cdot \delta_4(q_0,q_1,g) = 0.3,$$
$$[r_A(f(g(a),b),q_3)] = \big([r_A(g(a),q_1)] \cdot [r_A(b,q_1)]$$
$$\cdot\delta_6((q_1,q_1),q_3,f)\big) + \big([r_A(g(a),q_2)] \cdot [r_A(b,q_1)]$$
$$\cdot\delta_5((q_2,q_1),q_3,f)\big) = 0.3,$$
$$[r_A(f(b,b),q_3)] = [r_A(b,q_1)] \cdot \delta_6((q_1,q_1),q_3,f) = 0.3.$$

Thus, $Q_c = \{q_0, q_1, q_3\}$ is the set of accessible states of A with threshold $c = 0.2$. By using an easy method, it can be shown that $\bar{Q} = \{q_1\}$. Also $B(A, c) = \{b, g(a), f(b, b), f(g(a), b), f(g(a), g(a)), f(b, g(a))\}$.

Now, in the following lemmas we obtain useful results that will be used in some decision problems.

Lemma 3.5. *If q is an accessible state of A with a threshold c, there exists $t \in \mathcal{T}(F)$ such that $[r_A(t, q)] > c$ and $Height(t) \leq |Q|$.*

Proof. Since q is an accessible state of A with threshold c, there exists $t' \in \mathcal{T}(F)$ such that $[r_A(t', q)] > c$. If $Height(t') \leq |Q|$, it is enough to set $t = t'$. Otherwise, let $Height(t') > |Q|$. Now in t' consider a path of length strictly greater than $|Q|$. Along this path, there are two positions p_1 and p_2 such that $p_1 < p_2$, $[r(p_1, q)] > 0$ and $[r(p_2, q)] > 0$ for some state q. Let u be the ground subterm of t' at position p_2, and u' the ground subterm of t' at position p_1. Thus, there exists a non-trivial context C' so that $u' = C'[u]$. Now define the context C such that $t' = C[C'[u]]$. It is obvious that, $[r_A(C[u], q)] > c$. Now if $Height(C[u]) \leq |Q|$ the proof is completed. Otherwise, by similar iterations we can find $t \in \mathcal{T}(F)$ such that $Height(t) \leq |Q|$ and $[r_A(t, q)] > c$. □

Corollary 3.6. *If q is an accessible state of A, there exists $t \in \mathcal{T}(F)$ such that $[r_A(t, q)] = 1$ and $Height(t) \leq |Q|$.*

Proof. By definition of accessible state, q is an accessible state of A with threshold c, for all $c \in \mathcal{L}\backslash\{1\}$. Consider $\{c_n | c_n = 1\rho l_n\}$, where $l_n \in \mathcal{L}\backslash\{1\}$ and $n \in \mathbb{N}$ (the set of positive integers). By Lemma 3.5, corresponding to c_n, there exists $t_n \in \mathcal{T}(F)$ such that $[r_A(t_n, q)] > c_n$ and $Height(t_n) \leq |Q| < \infty$. Since $|Q|$ is finite, the elements of $\{t_n | n \in \mathbb{N}\}$ are finite. Therefore, at least one element of this sequence iterates infinitely. Let t be the mentioned element. As a result, $[r_A(t, q)] > 1\rho l_i,\ \forall i \in \mathbb{N}$. Consequently, $[r_A(t, q)] = 1$ and $Height(t) \leq |Q|$. □

From Lemma 3.5 and Corollary 3.6 we have the following Corollaries:

Corollary 3.7. *The behaviour of A with a threshold c is non-empty if and only if there exists a term t in $L^c(A)$ with $Height(t) \leq |Q|$.*

Proof. The proof is straightforward. □

Corollary 3.8. *The behaviour of A with a threshold c is infinite if and only if there exists a term t in $L^c(A)$ with $|Q| < Height(t) \leq 2|Q|$.*

Proof. Let $L^c(A)$ be infinite and for each term t in $L^c(A)$, $Height(t) > 2|Q|$. Let z be the shortest term in $L^c(A)$ with $Height(z) > 2|Q|$. Since $Height(z) > 2|Q| > |Q|$, Lemma 3.5 implies that there exist a non-trivial context C', a context C and a term u such that $z = C[C'[u]]$ and for all $n \geq 0$, $C[C'^n[u]] \in L^c(A)$. Take $C[C'^0[u]] = C[u] \in L^c(A)$. Therefore, $C[u] > 2|Q|$. Thus, there exists a term $t = C[u] \in L^c(A)$, such that $|z| > |t| > 2|Q|$, which contradicts to the assumption.
Conversely, assume that there exists a term t in $L^c(A)$ with $|Q| < Height(t) \leq 2|Q|$ and also let $L^c(A)$ be finite. Since $Height(t) > |Q|$, Lemma 3.5 implies that there

exist a non-trivial context C', a context C and a term u such that $t = C[C'[u]]$ and for all $n \geq 0$, $C[C'^n[u]] \in L^c(A)$. Therefore, $L^c(A)$ is infinite, which contradicts to the assumption. □

Example 3.9. In Example 3.4, q_0, q_1, q_3 are accessible states of A with threshold 0.2, and we have:

$[r_A(a, q_0)] = 0.4 > 0.2$ and $\mathit{Height}(a) = 1 < |Q| = 4$,
$[r_A(b, q_1)] = 1 > 0.2$ and $\mathit{Height}(b) = 1 < |Q| = 4$,
$[r_A(f(b, b), q_3)] = 0.3 > 0.2$ and $\mathit{Height}(f(b, b)) = 2 < |Q| = 4$.

Since for all terms $t \in B(A, c) = L^c(A)$ we have $\mathit{Height}(t) \leq |Q|$, the behaviour of A with threshold 0.2 is finite.

4. A Reduction Algorithm for an $\mathcal{L}$-valued Tree Automaton

In this section, we propose a reduction algorithm for an $\mathcal{L}$-valued tree automaton. Here, the purpose of reduction is to reduce the number of states of an $\mathcal{L}$-valued tree automaton with no change in its behaviour and the accessibility of all its states. An $\mathcal{L}$-valued tree automaton A is said to be reduced with a threshold c, if all its states are accessible with threshold c. In the following theorem, the existence of the reduced form of an $\mathcal{L}$-valued tree automaton is shown.

Theorem 4.1. *Let $\bar{L}$ be recognizable with a threshold c. Then there exists a reduced $\mathcal{L}$-valued tree automaton with threshold c that accepts $\bar{L}$.*

Proof. Let $A = (Q, F, Q_f, \delta)$ be a given $\mathcal{L}$-valued tree automaton where, $L^c(A) = \bar{L}$. Define $A' = (Q', F, Q'_f, \delta')$ as follows:

$$Q'_f(q) = \begin{cases} Q_f(q) & if\ Q_f(q) > c \\ 0 & o.w. \end{cases}$$

and

$$\delta'((q_1, \ldots, q_n), q, \sigma) = \begin{cases} \delta((q_1, \ldots, q_n), q, \sigma) & if\ \delta((q_1, \ldots, q_n), q, \sigma) > c \\ 0 & o.w., \end{cases}$$

where $q, q_1, \cdots, q_n \in Q$ and $\sigma \in F$. It is clear that A' is an $\mathcal{L}$-valued tree automaton with threshold c. Now, we show that $L^c(A') = L^c(A)$. If the term language recognized by A with threshold c is non-empty, there exists $t \in L^c(A)$ and $q \in Q$ such that $[r_A(t, q)] > c$. Consequently, $q \in Q_c$ and $[r_{A'}(t, q)] > c$. Therefore, $t \in L^c(A')$. Conversely, if $t \in L^c(A')$, there exists $q \in Q'$ such that $[r_{A'}(t, q)] > c$. Obviously $q \in Q$ and $[r_A(t, q)] > c$, therefore, $t \in L^c(A)$. Hence, $L^c(A') = L^c(A)$. □

Now, we propose an algorithm whose output is a reduced $\mathcal{L}$-valued tree automaton A' with a threshold c, equivalent to a given $\mathcal{L}$-valued tree automaton A.

Algorithm 4.2. Reduction algorithm for an $\mathcal{L}$-valued tree automaton

Input: $\mathcal{L}$-valued tree automaton $A = (Q, F, Q_f, \delta)$ and $c \in \mathcal{L} \setminus \{1\}$;
Step 1: Set $\Delta_j = \{\delta_k : \delta_k((q_1, \cdots, q_j, \cdots, q_n), q_i, f) > c \text{ and } q_j \in Q\}$ and $S = \emptyset$;

Step 2: For each $\delta_k : \delta_k((q_1, \cdots, q_n), q_i, f) > c,\ n \geq 0$
begin
Set $arr[k]$ to m; // m is the number of states in the left hand side of δ_k (without repetition)
if $m = 0$ then
move q_i into S;
end
Step 3: For each $\delta_k \in \Delta_j$ such that $q_j \in S$
begin
$arr[k] = arr[k] - 1$;
if $arr[k] = 0$ then
move q_i into S;
end
Step 4: Set Q' to S;
Step 5: Set Q'_f to $\{Q_f(q_i)|\ Q_f(q_i) > c,\ q_i \in S\}$;
Step 6: Set δ' to $\{\delta((q_1, \cdots, q_n), q, f)|\delta((q_1, \cdots, q_n), q, f) > c$ and $q, q_1, \cdots, q_n \in S\}$;
Output: $\mathcal{L}$-valued tree automaton $A' = (Q', F, Q'_f, \delta')$.

In the following, we provide an example to clarify Algorithm 4.2.

Example 4.3. Let A be the $\mathcal{L}$-valued tree automaton given in Example 3.4 and $c = 0.2$. Now, applying Algorithm 4.2, we construct a reduced $\mathcal{L}$-valued tree automaton $A' = (Q', F, Q'_f, \delta')$ from A as follows:

Step 1: Set $\Delta_0 = \{\delta_4(q_0, q_1, g)\}$ and $\Delta_1 = \{\delta_6((q_1, q_1), q_3, f)\}$.

Step 2: We construct array "arr" as follows:

0	0	1	1	2	1

Since arr[1]=arr[2]=0, set $S = \{q_0, q_1\}$.

Step 3: Since $\delta_4(q_0, q_1, g) \in \Delta_0$ and $q_0 \in S$, $arr[4] = 0$. Thus move q_1 into S. Hence, S remains unchanged. Also, since $\delta_6((q_1, q_1), q_3, f) \in \Delta_1$ and $q_1 \in S$, $arr[6] = 0$. Thus move q_3 into S. Therefore, we have $S = \{q_0, q_1, q_3\}$.
Step 4: Set $Q' = \{q_0, q_1, q_3\}$.
Step 5: Set $Q'_f(q_1) = 0.4$ and $Q'_f(q_3) = 0.5$.
Step 6: Set $\delta'_1(q_0, a) = 0.4$, $\delta'_2(q_1, b) = 1$, $\delta'_4(q_0, q_1, g) = 0.3$ and $\delta'_6((q_1, q_1), q_3, f) = 0.3$.

Therefore, the $\mathcal{L}$-valued tree automaton $A' = (Q', F, Q'_f, \delta')$ is the reduced form of A.

Now, we consider the complexity of the algorithm. It is obvious that Steps 2 and 3 run in $O\big(\sum_{\delta((q_1,\cdots,q_n),q,f)>c}(m+1)\big)$. The time needed for Steps 4, 5 and 6 is $O(|Q| + \sum_{\delta((q_1,\cdots,q_n),q,f)>c} 1)$. Therefore we have the following theorem:

Theorem 4.4. *Reduction Algorithm 4.2 runs in* $O(\|A\|)$.

Polynomial time reduction algorithms for $\mathcal{L}$-valued tree automata were given in [21, 35, 36]. They were called minimization algorithms, but the term minimization is

not adequate because they actually can not find a minimal $\mathcal{L}$-valued tree automaton but they can only find a smaller $\mathcal{L}$-valued tree automaton. In fact, the above mentioned minimization algorithms reduce just the number of the states but not the size of the $\mathcal{L}$-valued tree automaton. Therefore, they are just state reduction algorithms. The mentioned reduction algorithms ([21, 35, 36]) are applicable only to deterministic and complete $\mathcal{L}$-valued tree automata. However, our proposed reduction algorithm is applicable to any $\mathcal{L}$-valued tree automaton and runs in a linear time. It is important to note that since the input $\mathcal{L}$-valued tree automaton A in Example 4.3 is not complete, the algorithms established in [21, 35, 36] can not obtain a corresponding reduced lattice-valued tree automaton.

Note that if we let $\mathcal{L} = \{0, 1\}$, the proposed algorithm becomes a linear time reduction algorithm for crisp tree automata which has better time complexity than the given polynomial time reduction algorithms for crisp tree automata in the literature [1, 3, 4, 12].

5. Some of Decision Problems and Their Time Complexities

In this section, we discuss some of decision problems for $\mathcal{L}$-valued tree automata with a threshold c. Let us recall that the words over a finite alphabet can be viewed as unary terms, therefore automata become special tree automata. Hence, the described decision problems extend the corresponding problems in lattice automata [28]. Moreover, if we let $\mathcal{L} = \{0, 1\}$, these concepts and results reduce to their crisp counterparts discussed in [12].
The first decision problem, we are going to consider, is the membership-value problem for an $\mathcal{L}$-valued tree automaton with a threshold c.

5.1. **Membership-value Problem.** Let term $t \in \mathcal{T}(F)$, $c \in \mathcal{L} \setminus \{1\}$ and an $\mathcal{L}$-valued tree automaton A be given. The question is: does A recognize t with threshold c?
Run A on input t. If the run ends in a final state q of A such that $Q_f(q) > c$, and $[r_A(t, q)] > c$, set answer to true; if $[r_A(t, q)] \leq c$ or $Q_f(q) \leq c$, set answer to false. Therefore, we have the following theorem:

Theorem 5.1. *The set*

$\{t \in \mathcal{T}(F) |$ *$\mathcal{L}$-valued tree automaton A recognizes input term t with a threshold c*$\}$

is a decidable behaviour with threshold c.

5.2. **Emptiness-value Problem.** The emptiness-value problem for $\mathcal{L}$-valued tree automata is defined as follows:
given an $\mathcal{L}$-valued tree automaton A and a $c \in \mathcal{L} \setminus \{1\}$, is the language $L^c(A)$ empty?
From Corollary 3.7 it can be seen that the minimal height of recognized terms with a threshold c is bounded by the number of states. So, as the problem of membership-value in $L^c(A)$ is decidable, emptiness-value is decidable, too. Of course, this theory does not provide a practical algorithm. We know that the construction of the set of accessible states of an $\mathcal{L}$-valued tree automaton A provides an efficient solution

of the emptiness-value problem. To get an effective algorithm, it is useful to notice that A accepts the empty set if and only if no final state is accessible. Therefore, in order to decide the emptiness-value problem for $\mathcal{L}$-valued tree automaton A, it suffices to construct the set of accessible states of A and verify whether it contains a final state. In other words, to decide the emptiness-value problem for $\mathcal{L}$-valued tree automaton A, it is enough to find an $\mathcal{L}$-valued tree automaton A_c corresponding to A using Algorithm 4.2. Since reducing an $\mathcal{L}$-valued tree automaton is done in $O(\|A\|)$, we conclude that emptiness-value problem for $\mathcal{L}$-valued tree automata is done in $O(\|A\|)$. Therefore, we have the following theorem:

Theorem 5.2. *It can be decided in linear time whether an $\mathcal{L}$-valued tree automaton is empty with a threshold c.*

5.3. **Finiteness-value Problem.** Given an $\mathcal{L}$-valued tree automaton A and a $c \in \mathcal{L} \setminus \{1\}$, is the language $L^c(A)$ finite?
Using the proof of Corollary 3.8, we can propose a procedure for testing infiniteness-value of an $\mathcal{L}$-valued tree automaton with a threshold c. As we can decide the infiniteness-value problem for $\mathcal{L}$-valued tree automata, we can also decide finiteness-value problem.

5.4. **The Intersection-value Problem.** The intersection-value problem for $\mathcal{L}$-valued tree automata is as follows:
given two $\mathcal{L}$-valued tree automata A_1 and A_2 and a $c \in \mathcal{L} \setminus \{1\}$, is there at least one term recognized by A_1 and A_2 with threshold c? In the following theorem we consider this problem:

Theorem 5.3. *The intersection-value problem for $\mathcal{L}$-valued tree automata with a threshold c is decidable.*

Proof. Let $A_i = (Q_i, F, Q_{f_i}, \delta_i), i = 1, 2$ be two $\mathcal{L}$-valued tree automata. Now, we construct an $\mathcal{L}$-valued product tree automaton $A = (Q, F, Q_f, \delta)$ where,

$$Q = Q_1 \times Q_2,$$

$$\forall (q_1, q_2) \in Q, Q_f(q_1, q_2) = Q_{f_1}(q_1) \cdot Q_{f_2}(q_2)$$

and $\forall (q_i, p_i) \in Q, 1 \leq i \leq m$,

$$\delta\Big(\big((q_1, p_1), \cdots, (q_m, p_m)\big), (q, p), f\Big) = \delta_1\big((q_1, \ldots, q_m), q, f\big) \cdot \delta_2\big((p_1, \ldots, p_m), p, f\big).$$

Now, we prove that $L^c(A) = L^c(A_1) \cap L^c(A_2)$. We have

$$L^c(A) = \big\{t \in \mathcal{T}(F) | \sum_{(q,p) \in Q_1 \times Q_2} [r_A((q, p), t)] > c\big\} =$$

$$\big\{t \in \mathcal{T}(F) | \sum_{(q,p) \in Q_1 \times Q_2} \big([r_{A_1}(q, t) \cdot r_{A_2}(p, t)]\big) > c\big\} =$$

$$\big\{t \in \mathcal{T}(F) | \sum_{q \in Q_1} [r_{A_1}(q, t)] > c\big\} \cap \big\{t \in \mathcal{T}(F) | \sum_{p \in Q_2} [r_{A_2}(p, t)] > c\big\} =$$

$$L^c(A_1) \cap L^c(A_2).$$

In other words:

$$L^c(A)(t) = L^c(A_1)(t) \cdot L^c(A_2)(t), \forall t \in \mathcal{T}(F).$$

Therefore, the class of $\mathcal{L}$-valued tree languages with a threshold c is closed under intersection. Now, if we test the nonemptiness-value problem for $\mathcal{L}$-valued product tree automaton A, we get an algorithm in $O(\|A_1\| \times \|A_2\|)$. □

5.5. **The Union-value Problem.** In the following theorem the union-value decision problem is considered.

Theorem 5.4. *The union-value problem for $\mathcal{L}$-valued tree automata with a threshold c is decidable.*

Proof. Given two $\mathcal{L}$-valued tree automata $A_i = (Q_i, F, Q_{f_i}, \delta_i), i = 1, 2$, where $Q_1 \cap Q_2 = \emptyset$, now we construct an $\mathcal{L}$-valued tree automaton $A = (Q, F, Q_f, \delta)$ where:

$$Q = Q_1 \cup Q_2,$$

$$Q_f(q) = \begin{cases} Q_{f_1}(q) & q \in Q_1, \\ Q_{f_2}(q) & q \in Q_2, \end{cases}$$

and

$$\delta((q_1, \cdots, q_n), q, \sigma) = \begin{cases} \delta_1((q_1, \cdots, q_n), q, \sigma) & if\ q, q_1, \cdots, q_n \in Q_1, \\ \delta_2((q_1, \cdots, q_n), q, \sigma) & if\ q, q_1, \cdots, q_n \in Q_2. \end{cases}$$

Clearly, $L^c(A) = L^c(A_1) \cup L^c(A_2)$, that is the class of $\mathcal{L}$-valued tree languages with a threshold c is closed under union. Therefore, the union-value problem is decidable and the proof is completed. □

5.6. **The Equivalence-value Problem.** The equivalence-value problem for $\mathcal{L}$-valued tree automata is as follows:
given two $\mathcal{L}$-valued tree automata A_1 and A_2, whether $\mathcal{L}$-valued tree languages recognized by A_1 and A_2 with a threshold c are the same. Here, we consider this problem:

Theorem 5.5. *The equivalence-value problem is decidable for $\mathcal{L}$-valued tree automata with a threshold c.*

Proof. Let L_1 and L_2 be the languages of the $\mathcal{L}$-valued tree automata A_1 and A_2 with a threshold c, respectively. Also, let $L = L^c(A_1) \cup L^c(A_2)$. For all $t \in L$, if $\big(L_1(t) \rho L_2(t)\big) \cdot \big(L_2(t) \rho L_1(t)\big) = 1$, we conclude that $L_1 = L_2$, otherwise, $L_1 \neq L_2$. □

6. **Conclusions**

We considered tree automata based on residuated lattice-valued logic. Some concepts such as response function and accessible states (with a threshold c) of an $\mathcal{L}$-valued tree automaton were defined. Moreover, a reduction algorithm for $\mathcal{L}$-valued tree automata was proposed. Moreover, the efficiency of the proposed algorithm was compared with some existing reduction algorithms in the literature [21, 35, 36]. Also, some decision problems including membership-value, emptiness-value, intersection-value, union-value and equivalence-value problems were considered. Also, we computed the time needed to check some of these decision problems

for an $\mathcal{L}$-valued tree automaton.
However, for future work we hope to consider topics such as tree automata based on quantum-valued logic, equivalence between two lattice-valued tree automata, and $\mathcal{L}$-valued universal tree automata.

Acknowledgements. The author would like to express her sincere thanks to the editor and anonymous referees for their useful suggestions which improved the quality of the paper.

References

[1] P. A. Abdulla, L. Holík, L. Kaati and T. Vojnar, *A uniform (bi-)simulation-based framework for reducing tree automata*, Electronic Notes in Theoretical Computer Science, **251(3)** (2009), 27-48.

[2] K. H. Abolpour and M. M. Zahedi, *Isomorphism between two BL-general fuzzy automata*, Soft Computing, **16(4)** (2012), 729-736.

[3] R. Almeida, *Reducing nondeterministic tree automata by adding transitions*, In: J. Bouda, L. Holík, J. Kofroň, J. Strejček and A. Rambousek (Eds.), 11th Doctoral Workshop on Mathematical and Engineering Methods in Computer Science (MEMICS 2016), Telc, Czech Republic, Electronic Proceedings in Theoretical Computer Science, **233** (2016), 35-51.

[4] R. Almeida, L. Holík and R. Mayr, *Reduction of nondeterministic tree automata*, In: M. Chechik and J. Raskin (Eds.), Tools and Algorithms for the Construction and Analysis of Systems, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, **9636** (2016), 717-735.

[5] G. Birkhoff, *Lattice Theory*, American Mathematical Society, Providence, 1984.

[6] S. Bozapalidis and O. L. Bozapalidoy, *Fuzzy term language recognizability*, Fuzzy Sets and Systems, **161** (2010), 716-734.

[7] I. Chajda and J. Krnávek, *Skew residuated lattices*, Fuzzy Sets and Systems, **222** (2013), 78-83.

[8] W. Cheng and Z. Mo, *Minimization algorithm of fuzzy finite automata*, Fuzzy Sets and Systems, **141** (2004), 439-448.

[9] M. Ćirić, A. Stamenković, J. Ignjatović and T. Petković, *Factorization of fuzzy automata*, In: E. Csuhaj-Varju and Z. Ésik (eds.), FCT 2007, Lecture Notes in Computer Science, **4639** (2007), 213-225.

[10] M. Ćirić, A. Stamenković, J. Ignjatović and T. Petković, *Fuzzy relation equations and reduction of fuzzy automata*, Journal of Computer and System Sciences, **76(7)** (2010), 609-633.

[11] L. C. Ciungu, *Classes of residuated lattices*, Annals of University of Craiova, Math. Comp. Sci. Ser., **33** (2006), 189-207.

[12] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, C. Loding, S. Tison and M. Tommasi, *Tree Automata: Techniques and Applications*, Available: http://tata.gforge.inria.fr., 2007.

[13] A. Degtyarev, Y. U. Gurevich, P. Narendran, M. Veanes and A. Voronkov, *Decidability and complexity of simultaneous rigid E-unification with one variable and related results*, Theoretical Computer Science, **243** (2000), 167-184.

[14] J. E. Doner, *Decidability of the weak second-order theory of two successors*, Notices Amer. Math. Soc., **12** (1965), 365-468.

[15] J. E. Doner, *Term acceptors and some of their applications*, Journal of Comput. and Syst. Sci., **4** (1970), 406-451.

[16] M. Droste, T. Stüber and H. Vogler, *Weighted finite automata over strong bimonoids*, Inform. Sci., **180(1)** (2010), 156-166.

[17] M. Droste and H. Vogler, *Weighted automata and multi-valued logics over arbitrary bounded lattices*, Theoretical Computer Science, **418** (2012), 14-36.

[18] Z. Ésik and G. Liu, *Fuzzy tree automata*, Fuzzy Sets and Systems, **158** (2007), 1450-1460.

[19] J. P. Gallagher and G. Puebla, *Abstract interpretation over non-deterministic finite tree automata for set-based analysis of logic programs*, In: S. Krishnamurthi and C. R. Ramakrishnan (Eds.), Practical Aspects of Declarative Languages, Lecture Notes in Computer Science, **2257** (2002), 243-261.
[20] F. Gécseg and M. Steinby, *Tree Automata*, Akademiai Kiado, Budapest, 1984.
[21] M. Ghorani and M. M. Zahedi, *Characterizations of complete residuated lattice-valued finite tree automata*, Fuzzy Sets and Systems, **199** (2012), 28-46.
[22] M. Ghorani and M. M. Zahedi, *Alternating regular tree grammars in the framework of lattice-valued logic*, Iranian Journal of Fuzzy Systems, **13(2)** (2016), 71-94.
[23] M. Ghorani and M. M. Zahedi, *Coding tree languages based on lattice-valued logic*, Soft Computing, **21(14)** (2017), 3815-3825.
[24] M. Ghorani, M. M. Zahedi and R. Ameri, *Algebraic properties of complete residuated lattice-valued tree automata*, Soft Computing, **16(10)** (2012), 1723-1732.
[25] R. Gilleron, S. Tison and M. Tommasi, *Solving systems of set constraints using tree automata*, Lecture Notes in Computer Science, **665** (1993), 505–514.
[26] J. E. Hopcroft, R. Motwani and J. D. Ullman, *Introduction to Automata Theory, Languages and Computation*, Addison-wesley, New York, 1979.
[27] J. Ignjatović, M. Ćirić and S. Bogdanović, *Determinization of fuzzy automata with membership values in complete residuated lattices*, Information Sciences, **178(1)** (2008), 164-180.
[28] O. Kupferman and Y. Lustig, *Lattice automata*, In: B. Cook andA. Podelski (Eds.), Proceedings of the 8th International Conference on Verification, Model Checking, and Abstract Interpretation, VMCAI 2007, Lecture Notes in Comput. Sci., **4349**, pp. 199-213, Springer, Berlin, Heidelberg, 2007.
[29] K. Lehmann and R. Peñaloza, *The complexity of computing the behaviour of lattice automata on infinite trees*, Theoretical Computer Science, **534** (2014), 53-68.
[30] H. X. Lei and Y. Li, *Minimization of states in automata theory based on finite lattice-ordered monoids*, Information Sciences, **177** (2007), 1413-1421.
[31] Y. M. Li, Y. L. Li and Z. Y. Ma, *Computation tree logic model checking based on possibility measures*, Fuzzy Sets and Systems, **262** (2015), 44-59.
[32] Y. M. Li and Z. Y. Ma, *Quantitative computational tree logic model checking based on generalized possibility measures*, IEEE Transactions on Fuzzy Systems, **23(6)** (2015), 2034-2047.
[33] Y. M. Li and W. Pedrycz, *Minimization of lattice finite automata and its application to the decomposition of lattice languages*, Fuzzy Sets and Systems, **158(13)** (2007), 1423-1436.
[34] Y. Li and Q. Wang, *The universal fuzzy automata*, Fuzzy Sets and Systems, **249** (2014), 27-48.
[35] S. Moghari and M. M. Zahedi, *Minimization of deterministic fuzzy tree automata*, Journal of Fuzzy Set Valued Analysis, **2014** (2014), 1-18.
[36] S. Moghari and M. M. Zahedi, *Similarity-based minimization of fuzzy tree automata*, J. Appl. Math. Comput., **50(1)** (2016), 417-436.
[37] S. Moghari, M. M. Zahedi and R. Amer, *New direction in fuzzy tree automata*, Iranian Journal of Fuzzy Systems, **8(5)** (2011), 59-68.
[38] J. N. Mordeson and D. S. Malik, Fuzzy Automata and Languages: Theory and Applications, Chapman & Hall CRC, London, Boca Raton, 2002.
[39] P. Y. Pan, Y. M. Li, Y. Z. Cao and Z. Y. Ma, *Model checking fuzzy computation tree logic*, Fuzzy Sets and Systems, **262** (2015), 60-77.
[40] H. Y. Pan, Y. M. Li, Y. Z. Cao and Z. Y. Ma, *Model checking computation tree logic over finite lattices*, Theoretical Computer Science, **612** (2016), 45-62.
[41] J. Pavelka, *On fuzzy logic II: enriched residuated lattices and semantics of propositional calculi*, Z. Math. Logik Grundlagen Math., **25** (1979), 119-134.
[42] D. W. Qiu, *Automata theory based on completed residuated lattice-valued logic (I)*, Science in China (Series F), **44** (2001), 419-429.
[43] D. W. Qiu, *Automata theory based on completed residuated lattice-valued logic (II)*, Science in China (Series F), **45** (2002), 442-452.

[44] D. W. Qiu, *Pumping lemma in automata theory based on complete residuated lattice-valued logic: a note*, Fuzzy Sets and Systems, **157** (2006), 2128-2138.
[45] E. S. Santos, *On reduction of maxmin machines*, J. Math. Anal. Appl., **37** (1972), 677-686.
[46] A. Stamenković, M. Ćirić and J. Ignjatović, *Reduction of fuzzy automata by means of fuzzy quasi-orders*, Information Sciences, **275** (2014), 168-198.
[47] J. W. Thatcher and J. B. Wright, *Generalized finite automata with an application to a decision problem of second-order logic*, Mathematical System Theory, **2** (1968), 57-82.
[48] L. Wu and D. W. Qiu, *Automata theory based on completed residuated lattice-valued logic: reduction and minimization*, Fuzzy Sets and Systems, **161** (2010), 1635-1656.
[49] H. Y. Xing and D. W. Qiu, *Pumping lemma in context-free grammar theory based on complete residuated lattice-valued logic*, Fuzzy Sets and Systems, **160** (2009), 1141-1151.
[50] H. Y. Xing, D. W. Qiu, F. C. Liu and Z. J. Fan, *Equivalence in automata theory based on complete residuated lattice-valued logic*, Fuzzy Sets and Systems, **158** (2007), 1407-1422.

Maryam Ghorani, Faculty of Mathematical Sciences, Shahrood University of Technology, Shahrood, Iran.
E-mail address: ghorani@shahroodut.ac.ir