Available online at www.sciencedirect.com

ScienceDirect

Fuzzy Sets and Systems ••• (••••) •••–•••

FUZZY sets and systems

www.elsevier.com/locate/fss

Further improvements of determinization methods for fuzzy finite automata ☆

Zorana Jančić, Ivana Micić, Jelena Ignjatović, Miroslav Ćirić *

University of Niš, Faculty of Sciences and Mathematics, Višegradska 33, 18000 Niš, Serbia

Received 30 July 2014; received in revised form 23 November 2015; accepted 26 November 2015

Abstract

In this paper we provide further improvements of determinization methods for fuzzy finite automata. These methods perform better than all previous determinization methods for fuzzy finite automata, developed by Bělohlávek [3], Li and Pedrycz [38], Ignjatović et al. [25], and Jančić et al. [33], in the sense that they produce smaller automata, while require the same computation time. The only exception is the Brzozowski type determinization algorithm developed recently by Jančić and Ćirić [34], which produces a minimal crisp-deterministic fuzzy automaton, but the algorithms created here can also be used within the Brzozowski type algorithm and improve its performance.
© 2015 Published by Elsevier B.V.

Keywords: Fuzzy automaton; Crisp-deterministic fuzzy automaton; Nerode automaton; Determinization; State reduction; Complete residuated lattice

1. Introduction

Many practical applications of automata require determinization, a procedure of converting a nondeterministic finite automaton to an equivalent deterministic finite automaton, or, in the case of fuzzy automata, a procedure of converting a fuzzy finite automaton to an equivalent crisp-deterministic fuzzy automaton. The standard determinization method is the subset construction, where a nondeterministic automaton with n states is converted to an equivalent deterministic automaton with up to 2^n states, whereas in the case of fuzzy finite automata the resulting crisp-deterministic fuzzy automaton can even be infinite. That is why the main research directions in this area are aimed at finding such methods which will mitigate the potential enormous growth of the number of states during the determinization. The natural idea is to combine the existing determinization and state reduction methods so that we reduce

☆ Research supported by Ministry of Education, Science and Technological Development of the Republic of Serbia, Grant No. 174013.
* Corresponding author. Tel.: +38118224492; fax: +38118533014.
E-mail addresses: zoranajancic329@gmail.com (Z. Jančić), ivanajancic84@gmail.com (I. Micić), jelena.ignjatovic@pmf.edu.rs (J. Ignjatović), miroslav.ciric@pmf.edu.rs (M. Ćirić).

http://dx.doi.org/10.1016/j.fss.2015.11.019
0165-0114/© 2015 Published by Elsevier B.V.

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

the number of states to determinization. However, here we combine these methods to provide two-in-one procedures that perform determinization and state reduction simultaneously.

A crisp-deterministic fuzzy automaton is a fuzzy automaton with exactly one crisp initial state and a deterministic transition function, and the fuzziness is entirely concentrated in the fuzzy set of terminal states. This kind of determinism was first studied by Bělohlávek [3], in the context of fuzzy finite automata over a complete distributive lattice, and Li and Pedrycz [38], in the context of fuzzy finite automata over a lattice-ordered monoid. Determinization algorithms that were provided there generalize the subset construction. Another algorithm, provided by Ignjatović et al. [25], also generalizes the subset construction, and for any input it generates a smaller crisp-deterministic fuzzy automaton than the algorithms from [3,38]. Since this crisp-deterministic fuzzy automaton can be alternatively constructed by means of the Nerode right congruence of the original fuzzy finite automaton, it was called in [27] the *Nerode automaton* of this fuzzy finite automaton. The Nerode automaton was constructed in [25] for fuzzy finite automata over a complete residuated lattice, and it was noted that the identical construction can also be applied in a more general context, for fuzzy finite automata over a lattice-ordered monoid, and even for weighted finite automata over a semiring. This construction was also transferred in [16] to weighted automata over strong bimonoids. The algorithm proposed by Jančić et al. in [33] produces a crisp-deterministic fuzzy automaton that is even smaller than the Nerode automaton. In the terminology introduced in this paper, Jančić et al. constructed the children automaton for the Nerode automaton of a given fuzzy finite automaton. Recently, Jančić and Ćirić [34] adapted the well-known Brzozowski's double reversal determinization algorithm to fuzzy automata. As in the case of ordinary nondeterministic automata, Brzozowski type determinization of a fuzzy finite automaton results in a minimal crisp-deterministic fuzzy automaton that is equivalent to the original fuzzy finite automaton. It was also shown that even if all previous determinization algorithms fail to build a finite crisp-deterministic fuzzy automaton, the Brzozowski type algorithm can produce a finite one.

In addition to the determinization, practical applications of automata often require state reduction, a procedure of converting a given automaton into an equivalent automaton with a smaller number of states. As the state minimization problem for fuzzy finite automata, as well as for nondeterministic ones, is computationally hard (PSPACE-complete [35,37,69]), it is not required that this equivalent automaton is minimal, but it is necessary that it is effectively computable. From different aspects, the state reduction for fuzzy automata was studied in [1,15,36,44,48,50,51,64,67], as well as in the books [46,49]. All algorithms provided there were motivated by the basic idea used in the minimization of ordinary de terministic automata, the idea of detecting and merging indistinguishable states, which boils down to computation of certain crisp equivalences on the set of states. A new approach to the state reduction was initiated in [20,59]. First, it was shown that better reductions of fuzzy finite automata can be achieved if fuzzy equivalences are used instead of ordinary equivalences, and even better if fuzzy quasi-orders are used. In addition, it was shown that the state reduction problem for fuzzy finite automata can be reduced to the problem of finding fuzzy quasi-orders that are solutions to a particular system of fuzzy relation equations, called the general system. As the general system is difficult to solve, the problem was further reduced to the search for instances of the general system and their solutions which ensure the best possible reductions and can be efficiently computed. Two such instances, whose solutions were called right and left invariant, have the greatest solutions that can be computed in polynomial time, and two others, whose solutions were called weakly right and left invariant, have the greatest solutions that ensure better reductions, but their computation requires exponential time. For information about the reduction of the number of states of ordinary nondeterministic automata using right and left invariant equivalences and quasi-orders we refer to articles [9,10,28–31], as well as to articles [17,20,59] where relationships between the state reduction of fuzzy and nondeterministic automata are explained in detail.

The main aim of this paper is to combine determinization and state reduction methods into two-in-one algorithms that simultaneously perform determinization and state reduction. These algorithms perform better than all previous determinization algorithms for fuzzy finite automata, developed in [3,25,33,38], in the sense that they produce smaller automata, while require the same computation time. The only exception is the Brzozowski type determinization algorithm developed recently in [34], which produces a minimal crisp-deterministic fuzzy automaton, but we will see that the algorithms created here can be used within the Brzozowski type algorithm and improve its performance.

Our main results are the following. For any fuzzy finite automaton $\mathcal{A}$ and a reflexive weakly right invariant fuzzy relation φ on $\mathcal{A}$, we construct a crisp-deterministic fuzzy automaton $\mathcal{A}_\varphi$ and prove that it is equivalent to $\mathcal{A}$. If φ is a weakly right invariant fuzzy quasi-order, we show that the same automaton $\mathcal{A}_\varphi$ would be produced if we first perform the state reduction of $\mathcal{A}$ by means of φ and then construct the Nerode automaton of this reduced automaton. Furthermore, we show that automata $\mathcal{A}_\varphi$ determined by right invariant fuzzy quasi-orders are smaller than the Nerode

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

automaton of $\mathcal{A}$, and that larger right invariant fuzzy quasi-orders determine smaller crisp-deterministic fuzzy automata. For a fuzzy finite automaton $\mathcal{A}$ and a reflexive weakly right invariant fuzzy relation φ on $\mathcal{A}$, we also introduce the concept of the children automaton of $\mathcal{A}_\varphi$ and prove that it is equivalent to $\mathcal{A}$. In addition, if φ is a right invariant fuzzy quasi-order, we prove that the children automaton of $\mathcal{A}_\varphi$ is smaller than $\mathcal{A}_\varphi$, and that larger right invariant fuzzy quasi-orders determine smaller children automata. We also show that weakly left invariant fuzzy quasi-orders play a completely different role in the determinization. Namely, if they are used to reduce the number of states prior the construction of the Nerode automaton, they will be unsuccessful because the Nerode automaton of the reduced fuzzy automaton would be the same as the Nerode automaton of the original one. However, we show that weakly left invariant fuzzy quasi-orders may be successful in combination with the construction of the reverse Nerode automaton and that a two-in-one algorithm can be provided which can improve performance of the Brzozowski type algorithm for fuzzy finite automata.

It should be noted that some related issues have already been discussed in [62,63], in the context of ordinary nondeterministic automata. There van Glabbeek and Ploeger discussed five basic types of determinization algorithms for nondeterministic automata. Except the classical (accessible) *subset construction* they also discussed determinization using *transition sets*, *closures*, *simulation quasi-orders* and *compressions*. Here we exploit an idea similar to that which lies behind determinization using simulation quasi-orders, but there are some significant differences compared to the results from [62,63]. In the fuzzy framework, a direct generalization of simulation quasi-orders are right invariant fuzzy quasi-orders. However, our determinization method uses not only this kind of fuzzy quasi-orders, but also more general weakly right invariant fuzzy quasi-orders, which, in general, give better results in determinization than right invariant ones. We also work with left invariant and weakly left invariant fuzzy quasi-orders and show that they can also be used in the determinization. Admittedly, we show that during the determinization they do not reduce the number of states, but we also show that they can improve performances of those determinization methods that include determinization of the reverse fuzzy automaton (the Brzozowski type determinization [34] and the determinization by means of the degrees of language inclusion [45]). Another method that we use here, the construction of the so-called children automaton, is related to the determinization using transition sets that was discussed in [62,63]. In contrast to [62,63], where a similar construction was used only in combination with the subset construction, our construction can improve any other determinization method whose output is represented by a transition tree.

The structure of the paper is as follows. In Section 2 we recall basic notions and notation concerning fuzzy sets and relations, fuzzy automata and languages and crisp-deterministic fuzzy automata, we recall the concepts of the Nerode automaton and the reverse Nerode automaton, as well as the concepts of right and left invariant and weakly right and left invariant fuzzy relations. Our main theoretical results are presented in Section 3, and in Section 4 we provide algorithms, perform the analysis of their computation time, and give characteristic computational examples.

2. Preliminaries

2.1. Fuzzy sets and relations

In this paper we use complete residuated lattices as structures of membership values. A *residuated lattice* is an algebra $\mathcal{L} = (L, \wedge, \vee, \otimes, \to, 0, 1)$ such that

(L1) $(L, \wedge, \vee, 0, 1)$ is a lattice with the least element 0 and the greatest element 1,
(L2) $(L, \otimes, 1)$ is a commutative monoid with the unit 1,
(L3) $\otimes$ and $\to$ form an *adjoint pair*, i.e., they satisfy the *adjunction property*: for all $x, y, z \in L$,

$$x \otimes y \leqslant z \;\Leftrightarrow\; x \leqslant y \to z. \tag{1}$$

If, additionally, $(L, \wedge, \vee, 0, 1)$ is a complete lattice, then $\mathcal{L}$ is called a *complete residuated lattice*.

The algebra $(L, \vee, \otimes, 0, 1)$ is a semiring, and it is denoted by $\mathcal{L}^*$ and called the *semiring reduct* of $\mathcal{L}$.

The operations $\otimes$ (called *multiplication*) and $\to$ (called *residuum*) are intended for modeling the conjunction and implication of the corresponding logical calculus, and supremum ($\bigvee$) and infimum ($\bigwedge$) are intended for modeling of the existential and general quantifier, respectively. An operation $\leftrightarrow$ defined by

$$x \leftrightarrow y = (x \to y) \wedge (y \to x), \tag{2}$$

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

called *biresiduum* (or *biimplication*), is used for modeling the equivalence of truth values. For the basic properties of complete residuated lattices we refer to [2,4].

The most studied and applied structures of truth values, defined on the real unit interval $[0, 1]$ with $x \wedge y = \min(x, y)$ and $x \vee y = \max(x, y)$, are the *Łukasiewicz structure* ($x \otimes y = \max(x + y - 1, 0)$, $x \to y = \min(1 - x + y, 1)$), the *Goguen* (*product*) *structure* ($x \otimes y = x \cdot y$, $x \to y = 1$ if $x \leqslant y$ and $= y/x$ otherwise) and the *Gödel structure* ($x \otimes y = \min(x, y)$, $x \to y = 1$ if $x \leqslant y$ and $= y$ otherwise). Another important set of truth values is the set $\{a_0, a_1, \ldots, a_n\}$, $0 = a_0 < \cdots < a_n = 1$, with $a_k \otimes a_l = a_{\max(k+l-n,0)}$ and $a_k \to a_l = a_{\min(n-k+l,n)}$. A special case of the latter algebras is the two-element Boolean algebra of classical logic with the support $\{0, 1\}$. The only adjoint pair on the two-element Boolean algebra consists of the classical conjunction and implication operations. This structure of truth values is called the *Boolean structure*.

A partially ordered set P is said to satisfy the *descending chain condition* (briefly *DCC*) if every descending sequence of elements of P eventually terminates, i.e., if for every descending sequence $\{a_k\}_{k\in\mathbb{N}}$ of elements of P there is $k \in \mathbb{N}$ such that $a_k = a_{k+l}$, for all $l \in \mathbb{N}$. In other words, P satisfies DCC if there is no infinite descending chain in P.

In the sequel $\mathcal{L}$ will be a complete residuated lattice. A *fuzzy subset* of a set A *over* $\mathcal{L}$, or simply a *fuzzy subset* of A, is any mapping from A into L. Ordinary crisp subsets of A are considered as fuzzy subsets of A taking membership values in the set $\{0, 1\} \subseteq L$. Let f and g be two fuzzy subsets of A. The *equality* of f and g is defined as the usual equality of mappings, i.e., $f = g$ if and only if $f(x) = g(x)$, for every $x \in A$. The *inclusion* $f \leqslant g$ is also defined pointwise: $f \leqslant g$ if and only if $f(x) \leqslant g(x)$, for every $x \in A$. Endowed with this partial order the set L^A of all fuzzy subsets of A forms a complete residuated lattice, in which the meet (intersection) $\bigwedge_{i\in I} f_i$ and the join (union) $\bigvee_{i\in I} f_i$ of an arbitrary family $\{f_i\}_{i\in I}$ of fuzzy subsets of A are mappings from A into L defined by

$$\left(\bigwedge_{i\in I} f_i\right)(x) = \bigwedge_{i\in I} f_i(x), \qquad \left(\bigvee_{i\in I} f_i\right)(x) = \bigvee_{i\in I} f_i(x),$$

and the *product* $f \otimes g$ is a fuzzy subset defined by $f \otimes g(x) = f(x) \otimes g(x)$, for every $x \in A$.

A *fuzzy relation* between sets A and B (in this order) is any mapping from $A \times B$ to L, i.e., any fuzzy subset of $A \times B$, and the equality, inclusion (ordering), joins and meets of fuzzy relations are defined as for fuzzy sets. Set of all fuzzy relations between A and B will be denoted by $L^{A\times B}$. In particular, a fuzzy relation on a set A is any function from $A \times A$ to L, i.e., any fuzzy subset of $A \times A$. The set of all fuzzy relations on A will be denoted by $L^{A\times A}$. The *reverse* or *inverse* of a fuzzy relation $\alpha \in L^{A\times B}$ is a fuzzy relation $\alpha^{-1} \in L^{B\times A}$ defined by $\alpha^{-1}(b, a) = \alpha(a, b)$, for all $a \in A$ and $b \in B$. A crisp relation is a fuzzy relation which takes values only in the set $\{0, 1\}$, and if α is a crisp relation of A to B, then expressions "$\alpha(a, b) = 1$" and "$(a, b) \in \alpha$" will have the same meaning. The *crisp equality* on a set A is the crisp relation Δ_A on A defined by $\Delta_A(a, b) = 1$ if $a = b$, and $\Delta_A(a, b) = 0$, if $a \neq b$.

For non-empty sets A, B and C, and fuzzy relations $\alpha \in L^{A\times B}$ and $\beta \in L^{B\times C}$, their *composition* $\alpha \circ \beta \in L^{A\times C}$ is a fuzzy relation defined by

$$(\alpha \circ \beta)(a, c) = \bigvee_{b\in B} \alpha(a, b) \otimes \beta(b, c), \tag{3}$$

for all $a \in A$ and $c \in C$. For $f \in L^A$, $\alpha \in L^{A\times B}$ and $g \in L^B$, compositions $f \circ \alpha \in L^B$ and $\alpha \circ g \in L^A$ are fuzzy sets defined by

$$(f \circ \alpha)(b) = \bigvee_{a\in A} f(a) \otimes \alpha(a, b), \qquad (\alpha \circ g)(a) = \bigvee_{b\in B} \alpha(a, b) \otimes g(b), \tag{4}$$

for every $a \in A$ and $b \in B$. Finally, the composition of two fuzzy sets $f, g \in L^A$ is an element $f \circ g \in L$ (scalar) defined by

$$f \circ g = \bigvee_{a\in A} f(a) \otimes g(a). \tag{5}$$

When the underlying sets are finite, fuzzy relations can be interpreted as matrices and fuzzy sets as vectors with entries in L, and then the composition of fuzzy relations can be interpreted as the matrix product, compositions of fuzzy sets and fuzzy relations as vector-matrix products, and the composition of two fuzzy set as the scalar (dot) product.

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

It is easy to verify that the composition of fuzzy relations is associative, i.e.,

$$(\alpha \circ \beta) \circ \gamma = \alpha \circ (\beta \circ \gamma), \tag{6}$$

for all $\alpha \in L^{A\times B}$, $\beta \in L^{B\times C}$ and $\gamma \in L^{C\times D}$, and

$$(f \circ \alpha) \circ \beta = f \circ (\alpha \circ \beta), \quad (f \circ \alpha) \circ g = f \circ (\alpha \circ g), \quad (\alpha \circ \beta) \circ h = \alpha \circ (\beta \circ h) \tag{7}$$

for all $\alpha \in L^{A\times B}$, $\beta \in L^{B\times C}$, $f \in L^A$, $g \in L^B$ and $h \in L^C$. Hence, all parentheses in (6) and (7) can be omitted.

Let $\alpha, \beta \in L^{A\times A}$ and $f, g \in L^A$. The *right residual* of β by α is a fuzzy relation $\alpha\backslash\beta \in L^{A\times A}$ and the *left residual* of β by α is a fuzzy relation $\beta/\alpha \in L^{A\times A}$ defined by

$$(\alpha\backslash\beta)(a,b) = \bigwedge_{c\in A} \alpha(c,a) \to \beta(c,b), \qquad (\beta/\alpha)(a,b) = \bigwedge_{c\in A} \alpha(b,c) \to \beta(a,c), \tag{8}$$

whereas the *right residual* of g by f is a fuzzy relation $f\backslash g \in L^{A\times A}$ and the *left residual* of g by f is a fuzzy relation $g/f \in L^{A\times A}$ defined by

$$(f\backslash g)(a,b) = f(a) \to g(b), \qquad (g/f)(b,a) = f(a) \to g(b), \tag{9}$$

for all $a, b \in A$.

A fuzzy relation φ on a set A is said to be *reflexive*, if $\varphi(a,a) = 1$, to be *symmetric*, if $\varphi(a,b) = \varphi(b,a)$, and to be *transitive*, if $\varphi(a,b) \otimes \varphi(b,c) \leqslant \varphi(a,c)$, for all $a, b, c \in A$. A reflexive and transitive fuzzy relation is called a *fuzzy quasi-order* (in some sources *fuzzy preorder*). A symmetric fuzzy quasi-order is a *fuzzy equivalence*. For a fuzzy quasi-order φ on A and an element $a \in A$, the *φ-afterset* of a is a fuzzy set $a\varphi \in L^A$ defined by $a\varphi(b) = \varphi(a,b)$, and the *$\varphi$-foreset* of a is a fuzzy set $\varphi a \in L^A$ defined by $\varphi a(b) = \varphi(b,a)$, for every $b \in A$. When φ is interpreted as a matrix, then its aftersets are the rows, and its foresets are the columns of this matrix. If φ is a fuzzy equivalence, then the *φ-afterset* of a coincides with the *φ-foreset* of a, and it is called a *fuzzy equivalence class* of a.

2.2. Fuzzy automata

Throughout this paper, $\mathbb{N}$ denotes the set of natural numbers (without zero), X is an (finite) alphabet, X^+ and X^* denote, respectively, the free semigroup and the free monoid over X, ε denotes the empty word in X^*, and if not noted otherwise, $\mathcal{L}$ is a complete residuated lattice.

A *fuzzy automaton over $\mathcal{L}$* and X, or simply a *fuzzy automaton*, is a quadruple $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$, where A is a non-empty set, called the *set of states*, $\delta^A : A \times X \times A \to L$ is a fuzzy subset of $A \times X \times A$, called the *fuzzy transition function*, and $\sigma^A : A \to L$ and $\tau^A : A \to L$ are fuzzy subsets of A, called the *fuzzy set of initial states* and the *fuzzy set terminal states*, respectively. We can interpret $\delta^A(a,x,b)$ as the degree to which an input letter $x \in X$ causes a transition from a state $a \in A$ into a state $b \in A$, and we can interpret $\sigma^A(a)$ and $\tau^A(a)$ as the degrees to which a is respectively an input state and a terminal state. For methodological reasons we allow the set of states A to be infinite. A fuzzy automaton whose set of states is finite is called a *fuzzy finite automaton*. A fuzzy automaton over the Boolean structure is called a *nondeterministic automaton* or a *Boolean automaton*.

Define a family $\{\delta^A_x\}_{x\in X}$ of fuzzy relations on A by $\delta^A_x(a,b) = \delta^A(a,x,b)$, for each $x \in X$, and all $a, b \in A$, and extend this family to the family $\{\delta^A_u\}_{u\in X^*}$ inductively, as follows: $\delta^A_\varepsilon = \Delta_A$, where Δ_A is the crisp equality relation on A, and

$$\delta^A_{x_1x_2\ldots x_n} = \delta^A_{x_1} \circ \delta^A_{x_2} \circ \cdots \circ \delta^A_{x_n} \tag{10}$$

for all $n \in \mathbb{N}$, $x_1, x_2, \ldots, x_n \in X$. Members of this family are called *fuzzy transiton relations* of $\mathcal{A}$. Evidently, $\delta^A_{uv} = \delta^A_u \circ \delta^A_v$, for all $u, v \in X^*$. In addition, define families $\{\sigma^A_u\}_{u\in X^*}$ and $\{\tau^A_u\}_{u\in X^*}$ by

$$\sigma^A_u = \sigma^A \circ \delta^A_u, \qquad \tau^A_u = \delta^A_u \circ \tau^A, \tag{11}$$

for all $u \in X^*$.

We can visualize a fuzzy finite automaton $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ representing it as a labeled directed graph whose nodes are states of $\mathcal{A}$, an edge from a node a to a node b is labeled by pairs of the form $x/\delta^A_x(a,b)$, for any $x \in X$, and for any node a we draw an arrow labeled by $\sigma^A(a)$ that enters this node, and an arrow labeled by $\tau^A(a)$ coming

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

out of this node. For the sake of simplicity, we do not draw edges whose all labels are of the form $x/0$, and incoming and outgoing arrows labeled by 0. In particular, if $\mathcal{A}$ is a Boolean automaton, instead of any label of the form $x/1$ we write just x, initial states are marked by incoming arrows without any label, and terminal states are marked by double circles.

A *fuzzy language* in X^* over $\mathcal{L}$, or just a *fuzzy language*, is any fuzzy subset of X^*, i.e., any function from X^* into L. The *fuzzy language recognized by a fuzzy automaton* $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ is the fuzzy language $[\![\mathcal{A}]\!] \in L^{X^*}$ defined by

$$[\![\mathcal{A}]\!](u) = \bigvee_{a,b\in A} \sigma^A(a) \otimes \delta^A_u(a,b) \otimes \tau^A(b) = \sigma^A \circ \delta^A_u \circ \tau^A, \tag{12}$$

for any $u \in X^*$. In other words, the membership degree of the word u to the fuzzy language $[\![\mathcal{A}]\!]$ is equal to the degree to which $\mathcal{A}$ recognizes or accepts the word u. Fuzzy automata $\mathcal{A}$ and $\mathcal{B}$ are called *language equivalent*, or just *equivalent*, if $[\![\mathcal{A}]\!] = [\![\mathcal{B}]\!]$.

Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton over $\mathcal{L}$ and X and let φ be a fuzzy quasi-order on A. The fuzzy automaton $\mathcal{A}/\varphi = (A/\varphi, \sigma^{A/\varphi}, \delta^{A/\varphi}, \tau^{A/\varphi})$ where $A/\varphi = \{a\varphi \mid a \in A\}$ is the set of all φ-aftersets, the fuzzy transition function $\delta^{A/\varphi} : (A/\varphi) \times X \times (A/\varphi) \to L$ is given by

$$\delta^{A/\varphi}(a\varphi, x, b\varphi) = \bigvee_{a',b'\in A} \varphi(a,a') \otimes \delta^A_x(a',b') \otimes \varphi(b',b) = (\varphi \circ \delta^A_x \circ \varphi)(a,b), \tag{13}$$

for all $a, b \in A$ and $x \in X$, and the fuzzy set $\sigma^{A/\varphi} \in L^{A/\varphi}$ of initial states and the fuzzy set $\tau^{A/\varphi} \in L^{A/\varphi}$ of terminal states are defined by

$$\sigma^{A/\varphi}(a\varphi) = \bigvee_{a'\in A} \sigma^A(a') \otimes \varphi(a',a) = (\sigma^A \circ \varphi)(a), \quad a \in A \tag{14}$$

$$\tau^{A/\varphi}(a\varphi) = \bigvee_{a'\in A} \varphi(a,a') \otimes \tau^A(a') = (\varphi \circ \tau^A)(a), \quad a \in A \tag{15}$$

for all $a \in A$, is called the *afterset fuzzy automaton* of $\mathcal{A}$ with respect to φ. The *foreset fuzzy automaton* of $\mathcal{A}$ with respect to φ is defined dually, but since it is isomorphic to the afterset fuzzy automaton, we will work only with afterset fuzzy automata.

The cardinality of a fuzzy automaton $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$, in notation $|\mathcal{A}|$, is defined as the cardinality of its set of states A. A fuzzy automaton $\mathcal{A}$ is called *minimal fuzzy automaton* of a fuzzy language $f \in L^{X^*}$ if $[\![\mathcal{A}]\!] = f$ and $|\mathcal{A}| \leqslant |\mathcal{A}'|$, for every fuzzy automaton $\mathcal{A}'$ such that $[\![\mathcal{A}']\!] = f$. A minimal fuzzy automaton recognizing a given fuzzy language f is not necessarily unique up to an isomorphism. This is also true for nondeterministic automata.

Let $\mathcal{A} = (A, \delta^A, \sigma^A, \tau^A)$ be a fuzzy automaton over $\mathcal{L}$ and X. The *reverse fuzzy automaton* of $\mathcal{A}$ is a fuzzy automaton $\overline{\mathcal{A}} = (A, \bar{\delta}^A, \bar{\sigma}^A, \bar{\tau}^A)$, where $\bar{\sigma}^A = \tau^A$, $\bar{\tau}^A = \sigma^A$, and $\bar{\delta}^A : A \times X \times A \to L$ is defined by:

$$\bar{\delta}^A(a,x,b) = \delta^A(b,x,a),$$

for all $a, b \in A$ and $x \in X$. Roughly speaking, the reverse fuzzy automaton $\overline{\mathcal{A}}$ is obtained from $\mathcal{A}$ by exchanging fuzzy sets of initial and final states and "reversing" all the transitions. Due to the fact that the multiplication $\otimes$ is commutative, we have that $\bar{\delta}_u(a,b) = \delta_{\bar{u}}(b,a)$, for all $a, b \in A$ and $u \in X^*$.

The *reverse fuzzy language* of a fuzzy language $f \in L^{X^*}$ is a fuzzy language $\overline{f} \in L^{X^*}$ defined by $\overline{f}(u) = f(\bar{u})$, for each $u \in X^*$. As $\overline{(\bar{u})} = u$ for all $u \in X^*$, we have that $\overline{(\overline{f})} = f$, for any fuzzy language f. It is easy to see that the reverse fuzzy automaton $\overline{\mathcal{A}}$ recognizes the reverse fuzzy language $\overline{[\![\mathcal{A}]\!]}$ of the fuzzy language $[\![\mathcal{A}]\!]$ recognized by $\mathcal{A}$, i.e., $[\![\overline{\mathcal{A}}]\!] = \overline{[\![\mathcal{A}]\!]}$.

For more information on fuzzy automata over complete residuated lattices we refer to [20,59,24–26,52,53,64,66,27].

2.3. Crisp-deterministic fuzzy automata

Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton over X and $\mathcal{L}$. The fuzzy transition function δ^A is called *crisp-deterministic* if for every $x \in X$ and every $a \in A$ there exists $a' \in A$ such that $\delta^A_x(a,a') = 1$, and $\delta^A_x(a,b) = 0$, for all

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

$b \in A \setminus \{a'\}$. The fuzzy set of initial states σ^A is called *crisp-deterministic* if there exists $a_0 \in A$ such that $\sigma^A(a_0) = 1$, and $\sigma^A(a) = 0$, for every $a \in A \setminus \{a_0\}$. If both σ^A and δ^A are crisp-deterministic, then $\mathcal{A}$ is called a *crisp-deterministic fuzzy automaton* (for short: *cdfa*), and if it is finite, then it is called a *crisp-deterministic fuzzy finite automaton* (for short: *cdffa*).

A crisp-deterministic fuzzy automaton can also be defined as a quadruple $\mathcal{A} = (A, \delta^A, a_0, \tau^A)$, where A is a non-empty *set of states*, $\delta^A : A \times X \to A$ is a *transition function*, $a_0 \in A$ is an *initial state* and $\tau^A \in L^A$ is a *fuzzy set of terminal states*. The transition function δ^A can be extended to a function $\delta^A_* : A \times X^* \to A$ in the following way: $\delta^A_*(a, \varepsilon) = a$, for every $a \in A$, and $\delta^A_*(a, ux) = \delta^A(\delta^A_*(a, u), x)$, for all $a \in A$, $u \in X^*$ and $x \in X$. A state $a \in A$ is called *accessible* if there exists $u \in X^*$ such that $\delta^*(a_0, u) = a$. If every state of $\mathcal{A}$ is accessible, then $\mathcal{A}$ is called an *accessible crisp-deterministic fuzzy automaton*.

The initial state and transitions of a crisp-deterministic fuzzy automaton are graphically represented as in the case of Boolean automata, and the fuzzy set of terminal states is represented as in the case of fuzzy finite automata.

Let $\mathcal{A} = (A, \delta^A, a_0, \tau^A)$ and $\mathcal{B} = (B, \delta^B, b_0, \tau^B)$ be crisp-deterministic fuzzy automata. A function $\phi : A \to B$ is called a *homomorphism* of $\mathcal{A}$ into $\mathcal{B}$ if $\phi(a_0) = b_0$, $\phi(\delta^A(a, x)) = \delta^B(\phi(a), x)$ and $\tau^A(a) = \tau^B(\phi(a))$, for all $a \in A$ and $x \in X$. A bijective homomorphism is called an *isomorphism*. If there is a surjective homomorphism of $\mathcal{A}$ onto $\mathcal{B}$, then $\mathcal{B}$ is said to be a *homomorphic image* of $\mathcal{A}$, and if there is an isomorphism of $\mathcal{A}$ onto $\mathcal{B}$, then we say that $\mathcal{A}$ and $\mathcal{B}$ are *isomorphic crisp-deterministic fuzzy automata* and we write $\mathcal{A} \cong \mathcal{B}$.

The *language* of $\mathcal{A}$ is the fuzzy language $[\![\mathcal{A}]\!] : X^* \to L$ defined by

$$[\![\mathcal{A}]\!](u) = \tau(\delta^*(a_0, u)) \tag{16}$$

for every $u \in X^*$. Obviously, the image of $[\![\mathcal{A}]\!]$ is contained in the image of τ which is finite if the set of states A is finite. A fuzzy language $f : X^* \to L$ is called *cdffa-recognizable* if there is a crisp-deterministic fuzzy finite automaton $\mathcal{A}$ over X and $\mathcal{L}$ such that $[\![\mathcal{A}]\!] = f$. Then we say that $\mathcal{A}$ *recognizes* f.

A crisp-deterministic fuzzy automaton $\mathcal{A}$ is called a *minimal crisp-deterministic fuzzy automaton* of a fuzzy language f if $[\![\mathcal{A}]\!] = f$ and $|\mathcal{A}| < |\mathcal{A}'|$, for any crisp-deterministic fuzzy automaton $\mathcal{A}'$ such that $[\![\mathcal{A}']\!] = f$.

Next, the *Nerode automaton* of a fuzzy automaton $\mathcal{A} = (A, \sigma, \delta, \tau)$ is a crisp-deterministic fuzzy automaton $\mathcal{A}_N = (A_N, \sigma^A_\varepsilon, \delta_N, \tau_N)$, where $A_N = \{\sigma^A_u \mid u \in X^*\}$, and $\delta_N : A_N \times X \longrightarrow A_N$ and $\tau_N : A_N \to L$ are defined by

$$\delta_N(\sigma^A_u, x) = \sigma^A_{ux}, \qquad \tau_N(\sigma^A_u) = \sigma^A_u \circ \tau^A,$$

for every $u \in X^*$ and $x \in X$. The Nerode automaton was first constructed in [25], where it was shown that it is equivalent to the starting fuzzy automaton $\mathcal{A}$. The name "Nerode automaton" was introduced in [27].

The *reverse Nerode automaton* of a $\mathcal{A}$ is the Nerode automaton of the reverse fuzzy automaton of $\mathcal{A}$, i.e., the crisp-deterministic fuzzy automaton $\mathcal{A}_{\overline{N}} = (A_{\overline{N}}, \tau^A_\varepsilon, \delta_{\overline{N}}, \tau_{\overline{N}})$, where $A_{\overline{N}} = \{\tau^A_u \mid u \in X^*\}$, and $\delta_{\overline{N}} : A_{\overline{N}} \times X \to A_{\overline{N}}$ and $\tau_{\overline{N}} : A_{\overline{N}} \times L$ are defined by

$$\delta_{\overline{N}}(\tau^A_u, x) = \tau^A_{xu}, \qquad \tau_{\overline{N}}(\tau^A_u) = \sigma^A \circ \tau^A_u,$$

for all $u \in X^*$ and $x \in X$.

2.4. Right and left invariant fuzzy relations

In the rest of the paper, a fuzzy relation on a fuzzy automaton will mean a fuzzy relation on its set of states.

Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton. A fuzzy relation φ on A is a *right invariant* if

$$\varphi \circ \delta^A_x \leqslant \delta^A_x \circ \varphi, \qquad \text{for each } x \in X, \tag{17}$$

$$\varphi \circ \tau^A \leqslant \tau^A, \tag{18}$$

and it is called *weakly right invariant* if

$$\varphi \circ \tau^A_u \leqslant \tau^A_u, \qquad \text{for each } u \in X^*. \tag{19}$$

Similarly we define the dual concepts. A fuzzy relation φ on A is called *left invariant* if

$$\delta^A_x \circ \varphi \leqslant \varphi \circ \delta^A_x, \qquad \text{for each } x \in X, \tag{20}$$

$$\sigma^A \circ \varphi \leqslant \sigma^A, \tag{21}$$

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

and it is called *weakly left invariant* if

$$\sigma_u^A \circ \varphi \leqslant \sigma_u^A, \qquad \text{for each } u \in X^*. \tag{22}$$

It is easy to verify that every right invariant fuzzy relation is weakly right invariant, and every left invariant fuzzy relation is weakly left invariant. Note that if φ is reflexive, then φ satisfies (19) if and only if it satisfies

$$\varphi \circ \tau_u^A = \tau_u^A, \qquad \text{for each } u \in X^*, \tag{23}$$

and φ satisfies (22) if and only if it satisfies

$$\sigma_u^A \circ \varphi = \sigma_u^A, \qquad \text{for each } u \in X^*, \tag{24}$$

and consequently, φ satisfies (18) if and only if it satisfies $\varphi \circ \tau^A = \tau^A$, and it satisfies (21) if and only if it satisfies $\sigma^A \circ \varphi = \sigma^A$. Moreover, if φ is a fuzzy quasi-order, then φ satisfies (17) if and only if it satisfies

$$\varphi \circ \delta_x^A \circ \varphi = \delta_x^A \circ \varphi, \qquad \text{for each } x \in X, \tag{25}$$

and φ satisfies (20) if and only if it satisfies

$$\varphi \circ \delta_x^A \circ \varphi = \varphi \circ \delta_x^A, \qquad \text{for each } x \in X. \tag{26}$$

Right and left invariant fuzzy quasi-orders and fuzzy equivalences were introduced in [20,59], where they were used in the state reduction of fuzzy automata. They are closely related to forward and backward simulations and bisimulations between fuzzy automata, which were studied in [18,19]. Namely, a fuzzy quasi-order φ on a fuzzy automaton $\mathcal{A}$ is right invariant if its reverse φ^{-1} is a forward simulation of $\mathcal{A}$ into itself, and φ is left invariant if φ is a backward simulation of $\mathcal{A}$ into itself.

Weakly right and left invariant fuzzy quasi-orders were introduced in [59], and they were also used in the state reduction of fuzzy automata. They provide smaller automata than right invariant fuzzy quasi-orders, but are more difficult to compute. Weakly right and left invariant fuzzy quasi-orders and fuzzy equivalences are closely related to weak forward and backward simulations and bisimulations, which were studied in [32].

Algorithms for computing the greatest right and left invariant fuzzy quasi-orders on a fuzzy finite automaton, as well as algorithms for computing the greatest weakly right and left invariant ones, were provided in [59]. They are presented here in Section 4, together with an analysis of their computational time.

3. Theoretical results

Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton and φ a fuzzy relation on A. For each $u \in X^*$ we define a fuzzy set $\varphi_u : A \to L$ inductively, as follows: for the empty word ε and all $u \in X^*$ and $x \in X$ we set

$$\varphi_\varepsilon = \sigma^A \circ \varphi, \quad \varphi_{ux} = \varphi_u \circ \delta_x^A \circ \varphi \tag{27}$$

Clearly, if $u = x_1 \dots x_n$, where $x_1, \dots, x_n \in X$, then

$$\varphi_u = \sigma^A \circ \varphi \circ \delta_{x_1}^A \circ \varphi \circ \dots \circ \delta_{x_n}^A \circ \varphi. \tag{28}$$

Now, set $A_\varphi = \{\varphi_u \mid u \in X^*\}$, and define $\delta_\varphi : A_\varphi \times X \to A_\varphi$ and $\tau_\varphi : A_\varphi \to L$ as follows:

$$\delta_\varphi(\varphi_u, x) = \varphi_{ux}, \quad \tau_\varphi(\varphi_u) = \varphi_u \circ \tau^A, \tag{29}$$

for all $u \in X^*$ and $x \in X$. If $\varphi_u = \varphi_v$, for some $u, v \in X^*$, then for each $x \in X$ we have that

$$\delta_\varphi(\varphi_u, x) = \varphi_{ux} = \varphi_u \circ \delta_x \circ \varphi = \varphi_v \circ \delta_x \circ \varphi = \varphi_{vx} = \delta_\varphi(\varphi_v, x),$$

and hence, δ_φ is a well-defined function. It is clear that τ_φ is also a well-defined function, and consequently, $\mathcal{A}_\varphi = (A_\varphi, \varphi_\varepsilon, \delta_\varphi, \tau_\varphi)$ is a well-defined crisp-deterministic fuzzy automaton.

The main question that arises here is how to choose a fuzzy relation φ so that the automaton $\mathcal{A}_\varphi$ is equivalent to the original automaton $\mathcal{A}$. The following theorem gives an answer to this question.

Theorem 3.1. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton and φ a reflexive weakly right invariant fuzzy relation on $\mathcal{A}$. Then $\mathcal{A}_\varphi = (A_\varphi, \varphi_\varepsilon, \delta_\varphi, \tau_\varphi)$ is an accessible crisp-deterministic fuzzy automaton equivalent to $\mathcal{A}$.*

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

Proof. According to (23), by induction we easily prove that

$$\tau^A_{x_1x_2\ldots x_n} = \varphi \circ \delta^A_{x_1} \circ \varphi \circ \cdots \circ \delta^A_{x_n} \circ \varphi \circ \tau^A \tag{30}$$

for each $n \in \mathbb{N}$ and all $x_1, \ldots, x_n \in X$.

Now, according to (30), for each $u = x_1 \ldots x_n$, where $x_1, \ldots, x_n \in X$, we have that

$$[\![\mathcal{A}_\varphi]\!](u) = \tau_\varphi(\delta_\varphi(\varphi_\varepsilon, u)) = \tau_\varphi(\varphi_u) = \varphi_u \circ \tau^A = (\sigma^A \circ \varphi \circ \delta^A_{x_1} \circ \varphi \circ \ldots \circ \delta^A_{x_n} \circ \varphi) \circ \tau^A =$$
$$= \sigma^A \circ (\varphi \circ \delta^A_{x_1} \circ \varphi \circ \ldots \circ \delta^A_{x_n} \circ \varphi \circ \tau^A) = \sigma^A \circ \tau^A_u = \sigma^A \circ \delta^A_u \circ \tau^A = [\![\mathcal{A}]\!](u),$$

and besides,

$$[\![\mathcal{A}_\varphi]\!](\varepsilon) = \tau_\varphi(\varphi_\varepsilon) = \varphi_\varepsilon \circ \tau^A = \sigma^A \circ \varphi \circ \tau^A = \sigma^A \circ \tau^A = [\![\mathcal{A}]\!](\varepsilon),$$

which means that $[\![\mathcal{A}_\varphi]\!] = [\![\mathcal{A}]\!]$. Therefore, $\mathcal{A}_\varphi$ is equivalent to $\mathcal{A}$. □

In addition, the following is true.

Theorem 3.2. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton and φ a weakly right invariant fuzzy quasi-order on $\mathcal{A}$. Then the automaton $\mathcal{A}_\varphi$ is isomorphic to the Nerode automaton of the afterset fuzzy automaton $\mathcal{A}/\varphi$.*

Proof. For the sake of simplicity, set $B = A/\varphi$ and $\mathcal{B} = \mathcal{A}/\varphi$, i.e., let $\mathcal{B} = (B, \sigma^B, \delta^B, \tau^B)$ be the afterset fuzzy automaton of $\mathcal{A}$ corresponding to φ. Consider the Nerode automaton $\mathcal{B}_N = (B_N, \sigma^B_\varepsilon, \delta_N, \tau_N)$ of $\mathcal{B}$.

First, by induction on the length of a word, we will prove that for any $u \in X^*$ the following is true:

$$\sigma^B_u(a\varphi) = \varphi_u(a), \quad \text{for every } a \in A. \tag{31}$$

For any $a \in A$ we have that $\sigma^B_\varepsilon(a\varphi) = \sigma^B(a\varphi) = (\sigma^A \circ \varphi)(a) = \varphi_\varepsilon(a)$, and thus, (31) holds when u is the empty word. Next, suppose that (31) holds for some word $u \in X^*$. By (28) and idempotency of φ it follows that $\varphi_u \circ \varphi = \varphi_u$, so for each $x \in X$ and each $a \in A$ we have that

$$\sigma^B_{ux}(a\varphi) = (\sigma^B_u \circ \delta^B_x)(a\varphi) = \bigvee_{b\in A} \sigma^B_u(b\varphi) \otimes \delta^B_x(b\varphi, a\varphi) = \bigvee_{b\in A} \varphi_u(b) \otimes (\varphi \circ \delta^A_x \circ \varphi)(b, a) =$$
$$= (\varphi_u \circ \varphi \circ \delta^A_x \circ \varphi)(a) = (\varphi_u \circ \delta^A_x \circ \varphi)(a) = \varphi_{ux}(a).$$

Therefore, we conclude that (31) holds for every $u \in X^*$.

Now, define a function $\xi : A_\varphi \to B_N$ by $\xi(\varphi_u) = \sigma^B_u$, for each $u \in X^*$. For arbitrary $u, v \in X^*$ we have that

$$\varphi_u = \varphi_v \;\Leftrightarrow\; (\forall a \in A)\ \varphi_u(a) = \varphi_v(a) \;\Leftrightarrow\; (\forall a \in A)\ \sigma^B_u(a\varphi) = \sigma^B_v(a\varphi) \;\Leftrightarrow\; \sigma^B_u = \sigma^B_v,$$

so ξ is a well-defined and injective function. It is clear that ξ is also surjective, and therefore, ξ is a bijective function. Also, for all for all $u \in X^*$ and $x \in X$ we have that

$$\delta_N(\xi(\varphi_u), x) = \delta_N(\sigma^B_u, x) = \sigma^B_{ux} = \xi(\varphi_{ux}) = \xi(\delta_\varphi(\varphi_u, x)),$$
$$\tau_N(\xi(\varphi_u)) = \tau_N(\sigma^B_u) = \sigma^B_u \circ \tau^B = [\![\mathcal{B}]\!](u) = [\![\mathcal{A}]\!](u) = [\![\mathcal{A}_\varphi]\!](u) = \varphi_u \circ \tau^A = \tau_\varphi(\varphi_u),$$

so ξ is an isomorphism of the automaton $\mathcal{A}_\varphi$ onto the Nerode automaton $\mathcal{B}_N$ of $\mathcal{B} = \mathcal{A}/\varphi$. □

Remark 3.3. Note that in the previous theorem we need φ to be weakly right invariant only to prove that $\tau_N(\xi(\varphi_u)) = \tau_\varphi(\varphi_u)$. Everything else can be proved under the weaker assumption that φ is a fuzzy quasi-order.

In the case when working with right invariant fuzzy quasi-orders, it is possible to compare the size of the corresponding automata. This follows from the following theorem.

Theorem 3.4. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton and let φ and ϕ be right invariant fuzzy quasi-orders on $\mathcal{A}$ such that $\varphi \leqslant \phi$. Then the automaton $\mathcal{A}_\phi$ is a homomorphic image of the automaton $\mathcal{A}_\varphi$.*

Consequently, $|\mathcal{A}_\phi| \leqslant |\mathcal{A}_\varphi|$.

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

Proof. First we note that $\varphi \leqslant \phi$ is equivalent to $\varphi \circ \phi = \phi \circ \varphi = \phi$, because φ and ϕ are fuzzy quasi-orders.

Define a function $\xi : A_\varphi \to A_\phi$ by $\xi(\varphi_u) = \phi_u$, for each $u \in X^*$. First we prove that ξ is well-defined. Let $u, v \in X^*$ such that $\varphi_u = \varphi_v$. According to (25) and (28), by induction we easily prove that $\varphi_w = \sigma_w^A \circ \varphi$ and $\phi_w = \sigma_w^A \circ \phi$, for every $w \in X^*$, whence

$$\phi_u = \sigma_u^A \circ \phi = \sigma_u^A \circ \varphi \circ \phi = \varphi_u \circ \phi = \varphi_v \circ \phi = \sigma_v^A \circ \varphi \circ \phi = \sigma_v^A \circ \phi = \phi_v.$$

Therefore, ξ is a well-defined function. It is clear that ξ is a surjective function. Moreover, it is evident that $\delta_\phi(\xi(\varphi_u), x) = \xi(\delta_\varphi(\varphi_u, x))$, for all $u \in X^*$ and $x \in X$, and

$$\tau_\phi(\xi(\varphi_u)) = \tau_\phi(\phi_u) = \phi_u \circ \tau^A = [\![\mathcal{A}_\phi]\!](u) = [\![\mathcal{A}]\!](u) = [\![\mathcal{A}_\varphi]\!](u) = \varphi_u \circ \tau^A = \tau_\varphi(\varphi_u),$$

and hence, ξ is a homomorphism of $\mathcal{A}_\varphi$ onto $\mathcal{A}_\phi$ and $|\mathcal{A}_\phi| \leqslant |\mathcal{A}_\varphi|$. □

Note that when φ is a reflexive weakly left invariant fuzzy relation on a fuzzy automaton $\mathcal{A}$, then $\mathcal{A}_\varphi$ is just the Nerode automaton of $\mathcal{A}$, and we do not get any new construction. Besides, the following is true.

Theorem 3.5. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton and φ a weakly left invariant fuzzy quasi-order on $\mathcal{A}$. Then the Nerode automaton of the afterset fuzzy automaton $\mathcal{A}/\varphi$ is isomorphic to the Nerode automaton of $\mathcal{A}$.*

Proof. For the sake of simplicity, set $B = A/\varphi$ and $\mathcal{B} = \mathcal{A}/\varphi$, i.e., let $\mathcal{B} = (B, \sigma^B, \delta^B, \tau^B)$ be the afterset fuzzy automaton of $\mathcal{A}$ corresponding to φ.

First, by induction on the length of a word, we will prove that for any $u \in X^*$ the following is true:

$$\sigma_u^B(a\varphi) = \sigma_u^A(a), \quad \text{for every } a \in A. \tag{32}$$

For every $a \in A$ we have that $\sigma_\varepsilon^B(a\varphi) = (\sigma^A \circ \varphi)(a) = \sigma^A(a) = \sigma_\varepsilon^A(a)$, so (32) holds when u is the empty word. Next, suppose that (32) holds for some word $u \in X^*$. According to (32) and our starting hypothesis that φ is a weakly left invariant fuzzy quasi-order, for all $x \in X$ and $a \in A$ we obtain that

$$\begin{aligned}\sigma_{ux}^B(a\varphi) &= (\sigma_u^B \circ \delta_x^B)(a\varphi) = \bigvee_{b \in A} \sigma_u^B(b\varphi) \otimes \delta_x^B(b\varphi, a\varphi) = \bigvee_{b \in A} \sigma_u^A(b) \otimes (\varphi \circ \delta_x^A \circ \varphi)(b, a) \\ &= (\sigma_u^A \circ \varphi \circ \delta_x^A \circ \varphi)(a) = (\sigma_u^A \circ \delta_x^A \circ \varphi)(a) = (\sigma_{ux}^A \circ \varphi)(a) = \sigma_{ux}^A(a),\end{aligned}$$

what completes the proof of (32).

Now, define a function $\xi : B_N \to A_N$ by $\xi(\sigma_u^B) = \sigma_u^A$, for each $u \in X^*$. For arbitrary $u, v \in X^*$ we have that

$$\sigma_u^B = \sigma_v^B \;\Leftrightarrow\; (\forall a \in A)\ \sigma_u^B(a\varphi) = \sigma_v^B(a\varphi) \;\Leftrightarrow\; (\forall a \in A)\ \sigma_u^A(a) = \sigma_v^A(a) \;\Leftrightarrow\; \sigma_u^A = \sigma_v^A,$$

which means that ξ is a well-defined and injective function. In addition, ξ is surjective, and consequently, ξ is a bijective function. It is easy to check that ξ is a homomorphism, and therefore, ξ is an isomorphism of the Nerode automaton of $\mathcal{B}$ onto the Nerode automaton of $\mathcal{A}$. □

As we said earlier, when φ is a reflexive weakly left invariant fuzzy relation on a fuzzy automaton $\mathcal{A}$, then $\mathcal{A}_\varphi$ is just the Nerode automaton of $\mathcal{A}$, and we do not get any new construction. However, we will show that weakly left invariant fuzzy relations work well with another construction, and can be very useful in the determinization of the reverse fuzzy automaton of $\mathcal{A}$.

Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton and ψ a fuzzy relation on A. For each $u \in X^*$ we define a fuzzy set $\psi^u : A \to L$ inductively, as follows: for the empty word ε and all $u \in X^*$ and $x \in X$ we set

$$\psi^\varepsilon = \psi \circ \tau^A, \quad \psi^{xu} = \psi \circ \delta_x^A \circ \psi^u \tag{33}$$

Clearly, if $u = x_1 \dots x_n$, where $x_1, \dots, x_n \in X$, then

$$\psi^u = \psi \circ \delta_{x_1}^A \circ \psi \circ \dots \circ \delta_{x_n}^A \circ \psi \circ \tau^A. \tag{34}$$

Now, set $A^\psi = \{\psi_u \mid u \in X^*\}$, and define $\delta^\psi : A^\psi \times X \to A^\psi$ and $\tau^\psi : A^\psi \to L$ as follows:

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

$$\delta^{\psi}(\psi^u, x) = \psi^{xu}, \quad \tau^{\psi}(\psi^u) = \sigma^A \circ \psi^u, \tag{35}$$

for all $u \in X^*$ and $x \in X$. If $\psi_u = \psi_v$, for some $u, v \in X^*$, then for each $x \in X$ we have that

$$\delta^{\psi}(\psi^u, x) = \psi^{xu} = \psi \circ \delta_x^A \circ \psi^u = \psi \circ \delta_x^A \circ \psi^v = \psi^{xv} = \delta^{\psi}(\psi^v, x),$$

and hence, δ^{ψ} is a well-defined function. Clearly, τ^{ψ} is also a well-defined function, so $\mathcal{A}^{\psi} = (A^{\psi}, \psi^{\varepsilon}, \delta^{\psi}, \tau^{\psi})$ is a well-defined crisp-deterministic fuzzy automaton.

Now we prove that the following is true.

Theorem 3.6. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton and ψ a reflexive weakly left invariant fuzzy relation on $\mathcal{A}$. Then $\mathcal{A}^{\psi} = (A^{\psi}, \psi^{\varepsilon}, \delta^{\psi}, \tau^{\psi})$ is an accessible crisp-deterministic fuzzy automaton equivalent to the reverse fuzzy automaton of $\mathcal{A}$.*

Proof. Consider an arbitrary word $u = x_1 \dots x_n$, where $x_1, \dots, x_n \in X$. According to (24), by induction we obtain that

$$\sigma^A \circ \psi \circ \delta_{x_n}^A \circ \psi \circ \dots \circ \delta_{x_1}^A \circ \psi = \sigma_{x_n \dots x_1}^A,$$

whence it follows that

$$\begin{aligned}[\![\mathcal{A}^{\psi}]\!](u) &= \tau^{\psi}(\delta^{\psi}(\psi^{\varepsilon}, u)) = \tau^{\psi}(\psi^{\bar{u}}) = \sigma^A \circ \psi^{\bar{u}} = \sigma^A \circ (\psi \circ \delta_{x_n}^A \circ \psi \circ \dots \circ \delta_{x_1}^A \circ \psi \circ \tau^A) = \\ &= (\sigma^A \circ \psi \circ \delta_{x_n}^A \circ \psi \circ \dots \circ \delta_{x_1}^A \circ \psi) \circ \tau^A = \sigma_{x_n \dots x_1}^A \circ \tau^A = \sigma_{\bar{u}}^A \circ \tau_A = [\![\mathcal{A}]\!](\bar{u}) = [\![\overline{\mathcal{A}}]\!](u).\end{aligned}$$

On the other hand,

$$[\![\mathcal{A}^{\psi}]\!](\varepsilon) = \tau^{\psi}(\psi^{\varepsilon}) = \sigma^A \circ \psi^{\varepsilon} = \sigma^A \circ \psi \circ \tau^A = \sigma^A \circ \tau^A = [\![\mathcal{A}]\!](\varepsilon) = [\![\overline{\mathcal{A}}]\!](\varepsilon).$$

Therefore, $[\![\mathcal{A}^{\psi}]\!] = [\![\overline{\mathcal{A}}]\!]$, i.e., $\mathcal{A}^{\psi}$ is equivalent to $\overline{\mathcal{A}}$. □

The next two theorems can be proved similarly as Theorems 3.2 and 3.4, so their proofs will be omitted.

Theorem 3.7. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton and ψ a weakly left invariant fuzzy quasi-order on $\mathcal{A}$. Then the automaton $\mathcal{A}^{\psi}$ is isomorphic to the reverse Nerode automaton of the afterset fuzzy automaton $\mathcal{A}/\psi$.*

Theorem 3.8. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton and let ψ and ψ' be left invariant fuzzy quasi-orders on $\mathcal{A}$ such that $\psi \leqslant \psi'$. Then the automaton $\mathcal{A}^{\psi'}$ is a homomorphic image of the automaton $\mathcal{A}^{\psi}$.*

Note that the reverse Nerode automaton plays a crucial role in Brzozowski type determinization of a fuzzy automaton. Namely, it has been proven in [34] that when we start from a fuzzy automaton $\mathcal{A}$, two consecutive applications of the construction of a reverse Nerode automaton produce a minimal crisp-deterministic fuzzy automaton which is equivalent to $\mathcal{A}$. We will show here that the first of these two constructions can be replaced by construction of the automaton $\mathcal{A}^{\psi}$, for some reflexive weakly left invariant fuzzy relation ψ on $\mathcal{A}$.

Theorem 3.9. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton and ψ a reflexive weakly left invariant fuzzy relation on $\mathcal{A}$. Then the reverse Nerode automaton of $\mathcal{A}^{\psi}$ is a minimal crisp-deterministic fuzzy automaton equivalent to $\mathcal{A}$.*

Proof. As we have proved in Theorem 3.6, $\mathcal{A}^{\psi}$ is an accessible crisp-deterministic fuzzy automaton equivalent to $\overline{\mathcal{A}}$. According to Theorem 3.5 [34], for any accessible crisp-deterministic fuzzy automaton $\mathcal{B}$, the reverse Nerode automaton of $\mathcal{B}$ is a minimal crisp-deterministic fuzzy automaton equivalent to $\overline{\mathcal{B}}$. Therefore, $\mathcal{A}^{\psi}$ is a minimal crisp-deterministic fuzzy automaton equivalent to the reverse of $\overline{\mathcal{A}}$, i.e., it is a minimal crisp-deterministic fuzzy automaton equivalent to $\mathcal{A}$. □

As the automaton $\mathcal{A}^{\psi}$ can be significantly smaller than the reverse Nerode automaton $\mathcal{A}_{\overline{N}}$ (in particular, if ψ is the greatest left invariant fuzzy quasi-order on $\mathcal{A}$), replacing $\mathcal{A}_{\overline{N}}$ with $\mathcal{A}^{\psi}$ in the first step of Brzozowski type procedure we could mitigate a combinatorial blow up of the number of states that may happen in this step. In the second step, such

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

a problem does not exist because both constructions give the minimal crisp-deterministic fuzzy automaton equivalent to $\mathcal{A}$.

Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton over an alphabet $X = \{x_1, \ldots, x_m\}$ and φ a fuzzy relation on A. For each $u \in X^*$ define an $(m+1)$-tuple φ_u^c by

$$\varphi_u^c = (\varphi_{ux_1}, \ldots, \varphi_{ux_m}, \varphi_u \circ \tau^A) = (\varphi_u \circ \delta_{x_1}^A, \ldots, \varphi_u \circ \delta_{x_m}^A, \varphi_u \circ \tau^A),$$

set $A_\varphi^c = \{\varphi_u^c \mid u \in X^*\}$, and define $\delta_\varphi^c : A_\varphi^c \times X \to A_\varphi^c$ and $\tau_\varphi^c : A_\varphi^c \to L$ as follows:

$$\delta_\varphi^c(\varphi_u^c, x) = \varphi_{ux}^c, \qquad \tau_\varphi^c(\varphi_u^c) = \varphi_u \circ \tau^A, \tag{36}$$

for all $u \in X^*$ and $x \in X$. We have the following:

Theorem 3.10. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton over an alphabet $X = \{x_1, \ldots, x_m\}$ and φ a reflexive weakly right invariant fuzzy relation on A. Then $\mathcal{A}_\varphi^c = (A_\varphi^c, \varphi_\varepsilon^c, \delta_\varphi^c, \tau_\varphi^c)$ is an accessible crisp-deterministic fuzzy automaton equivalent to $\mathcal{A}$.*

Proof. Let $\varphi_u^c = \varphi_v^c$, for some $u, v \in X^*$. This means that $\varphi_{ux_i} = \varphi_{vx_i}$, for any $i \in \{1, \ldots, m\}$, and $\varphi_u \circ \tau^A = \varphi_v \circ \tau^A$. Now, for an arbitrary $x \in X$ we have that $\varphi_{ux} = \varphi_{vx}$, whence

$$\varphi_{uxx_i} = \varphi_{ux} \circ \delta_{x_i}^A \circ \varphi = \varphi_{vx} \circ \delta_{x_i}^A \circ \varphi = \varphi_{vxx_i},$$

for each $i \in \{1, \ldots, m\}$, and also, $\varphi_{ux} \circ \tau^A = \varphi_{vx} \circ \tau^A$. Hence, $\varphi_{ux}^c = \varphi_{vx}^c$, so $\delta_\varphi^c(\varphi_u^c, x) = \varphi_{ux}^c = \varphi_{vx}^c = \delta_\varphi^c(\varphi_v^c, x)$, and this means that δ_φ^c is a well-defined function. Clearly, τ_φ^c is also a well-defined function, and consequently, $\mathcal{A}_\varphi^c$ is an accessible crisp-deterministic automaton.

Next, for each $u \in X^*$ we have that

$$[\![\mathcal{A}_\varphi^c]\!](u) = \tau_\varphi^c(\delta_\varphi^c(\varphi_\varepsilon^c, u)) = \tau_\varphi^c(\varphi_u^c) = \varphi_u \circ \tau^A = [\![\mathcal{A}_\varphi]\!](u) = [\![\mathcal{A}]\!](u),$$

and therefore, $[\![\mathcal{A}_\varphi^c]\!]$ is equivalent to $\mathcal{A}$. □

In the case when φ is the crisp equality on A, we have that $\varphi_u = \sigma_u^A$, for each $u \in X^*$, and $\mathcal{A}_\varphi^c$ is the automaton constructed in [33], where it was called the *reduced Nerode automaton* of $\mathcal{A}$. Here we will use a different terminology. Namely, the first m elements in the $(m+1)$-tuple φ_u^c are the children of the vertex φ_u in the transition tree of the automaton $\mathcal{A}_\varphi$ (see Algorithm 4.1), and the $m+1$st element of φ_u^c is the termination degree of φ_u in $\mathcal{A}_\varphi$. For this reason, the automaton $\mathcal{A}_\varphi^c$ will be called the *children automaton* of $\mathcal{A}_\varphi$. In this regard, the above mentioned automaton constructed in [33] is the children automaton of the Nerode automaton $\mathcal{A}_N$ of $\mathcal{A}$. The children automaton of the Nerode automaton $\mathcal{A}_N$ is denoted by $\mathcal{A}_N^c = (A_N^c, \sigma_\varepsilon^c, \delta_N^c, \tau_N^c)$.

Theorem 3.11. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton over an alphabet $X = \{x_1, \ldots, x_m\}$ and let φ and ϕ be right invariant fuzzy quasi-orders on A such that $\varphi \leqslant \phi$. Then the automaton $\mathcal{A}_\phi^c$ is a homomorphic image of the automaton $\mathcal{A}_\varphi^c$, and consequently, $|\mathcal{A}_\phi^c| \leqslant |\mathcal{A}_\varphi^c|$.*

Proof. This theorem can be proved analogously as Theorem 3.4, so the proof will be omitted. □

Theorem 3.12. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton over an alphabet $X = \{x_1, \ldots, x_m\}$ and let φ be a weakly right invariant fuzzy quasi-order on $\mathcal{A}$. Then the automaton $\mathcal{A}_\varphi^c$ is isomorphic to the automaton $\mathcal{B}_N^c$, where $\mathcal{B} = \mathcal{A}/\varphi$ is the afterset fuzzy automaton of $\mathcal{A}$ with respect to φ.*

Proof. Define a function $\xi : A_\varphi^c \to B_N^c$ by $\xi(\varphi_u^c) = \sigma_u^{B,c} = (\sigma_{ux_1}^B, \ldots, \sigma_{ux_m}^B, \sigma_u^B \circ \tau^B)$, for each $u \in X^*$. As in the proof of Theorem 3.2 we obtain that $\sigma_u^B(a\varphi) = \varphi_u(a)$, for all $u \in X^*$ and $a \in A$, and by this it follows that

$$\varphi_{ux_i} = \varphi_{vx_i} \;\Leftrightarrow\; (\forall a \in A)\ \varphi_{ux_i}(a) = \varphi_{vx_i}(a) \;\Leftrightarrow\; (\forall a \in A)\ \sigma_{ux_i}^B(a\varphi) = \sigma_{vx_i}^B(a\varphi) \;\Leftrightarrow\; \sigma_{ux_i}^B = \sigma_{vx_i}^B,$$

and also,

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

$$\varphi_u \circ \tau^A = \varphi_v \circ \tau^A \;\Leftrightarrow\; [\![\mathcal{A}_\varphi]\!](u) = [\![\mathcal{A}_\varphi]\!](v) \;\Leftrightarrow\; [\![\mathcal{A}]\!](u) = [\![\mathcal{A}]\!](v)$$
$$\Leftrightarrow\; [\![\mathcal{B}]\!](u) = [\![\mathcal{B}]\!](v) \;\Leftrightarrow\; [\![\mathcal{B}_N]\!](u) = [\![\mathcal{B}_N]\!](v) \;\Leftrightarrow\; \sigma_u^B \circ \tau^B = \sigma_v^B \circ \tau^B,$$

for all $u, v \in X^*$ and $x_i \in X$, and therefore, $\varphi_u^c = \varphi_v^c$ if and only if $\sigma_u^{B,c} = \sigma_v^{B,c}$. This means that ξ is a well-defined and injective function. It is clear that ξ is also surjective, and it can be easily verified that it is a homomorphism. Hence, ξ is an isomorphism of $\mathcal{A}_\varphi^c$ onto $\mathcal{B}_N^c$. □

Theorem 3.13. *Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy automaton over an alphabet $X = \{x_1, \ldots, x_m\}$ and let φ be a weakly right invariant fuzzy quasi-order on A. Then the automaton $\mathcal{A}_\varphi^c$ is a homomorphic image both of $\mathcal{A}_\varphi$ and $\mathcal{A}_N^c$, and consequently, $|\mathcal{A}_\varphi^c| \leqslant |\mathcal{A}_\varphi|$ and $|\mathcal{A}_\varphi^c| \leqslant |\mathcal{A}_N^c|$.*

Proof. According to Theorems 3.2 and 3.12, $\mathcal{A}_\varphi$ is isomorphic to $\mathcal{B}_N$, and $\mathcal{A}_\varphi^c$ is isomorphic to $\mathcal{B}_N^c$, and by Theorem 3.4 [33], $\mathcal{B}_N^c$ is a homomorphic image of $\mathcal{B}_N$. Therefore, $\mathcal{A}_\varphi^c$ is a homomorphic image of $\mathcal{A}_\varphi$.

On the other hand, $\mathcal{A}_N^c$ is isomorphic to $\mathcal{A}_\Delta^c$, where Δ is the crisp equality on A, and by Theorem 3.11, $\mathcal{A}_\varphi^c$ is a homomorphic image of $\mathcal{A}_N^c$. □

4. Algorithms and computational examples

Let $c_\vee$, $c_\wedge$, $c_\otimes$ and $c_\to$ be respectively computation times of the operations $\vee$, $\wedge$, $\otimes$ and $\to$ in $\mathcal{L}$. In particular, if $\mathcal{L}$ is linearly ordered, we can assume that $c_\vee = c_\wedge = 1$, and when $\mathcal{L}$ is the Gödel structure, we can also assume that $c_\otimes = c_\to = 1$.

Algorithm 4.1 *(Construction of the automaton $\mathcal{A}_\varphi$).* The input of this algorithm are a fuzzy finite automaton $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ with n states, over a finite alphabet X with m letters, and a fuzzy relation φ on A, and the output is the crisp-deterministic fuzzy automaton $\mathcal{A}_\varphi = (A_\varphi, \varphi_\varepsilon, \delta_\varphi, \tau_\varphi)$.

The procedure is to construct the *transition tree* of $\mathcal{A}_\varphi$ directly from $\mathcal{A}$, and during this procedure we use pointers $s(\cdot)$ which points vertices of the tree under construction to the corresponding integers. The transition tree of $\mathcal{A}_\varphi$ is constructed inductively as follows:

(A1) The root of the tree is $\varphi_\varepsilon = \sigma^A \circ \varphi$, and we put $T_0 = \{\varphi_\varepsilon\}$ and $s(\varphi_\varepsilon) = 1$, and we compute the value $\tau_\varphi(\varphi_\varepsilon) = \varphi_\varepsilon \circ \tau^A$.

(A2) After the ith step let a tree T_i have been constructed, and vertices in T_i have been labeled either 'closed' or 'non-closed'. The meaning of these two terms will be made clear in the sequel.

(A3) In the next step we construct a tree T_{i+1} by enriching T_i in the following way: for any non-closed leaf φ_u occurring in T_i, where $u \in X^*$, and any $x \in X$ we add a vertex $\varphi_{ux} = \varphi_u \circ \delta_x^A \circ \varphi$ and an edge from φ_u to φ_{ux} labelled by x. Simultaneously, we check whether φ_{ux} is a fuzzy set that has already been constructed. If it is true, if φ_{ux} is equal to some previously computed φ_v, we mark φ_{ux} as closed and set $s(\varphi_{ux}) = s(\varphi_v)$. Otherwise, we compute the value $\tau_\varphi(\varphi_{ux}) = \varphi_{ux} \circ \tau^A$ and set $s(\varphi_{ux})$ to be the next unassigned integer. The procedure terminates when all leaves are marked closed.

(A4) When the transition tree of $\mathcal{A}_\varphi$ is constructed, we erase all closure marks and glue leaves to interior vertices with the same pointer value. The diagram that results is the transition graph of $\mathcal{A}_\varphi$.

When φ is taken to be the crisp equality on A, Algorithm 4.1 gives the Nerode automaton $\mathcal{A}_N$ of $\mathcal{A}$.

The above described procedure does not necessarily terminate in a finite number of steps, since the collection $\{\varphi_u\}_{u \in X^*}$ may be infinite. However, in cases when this collection is finite, the procedure will terminate in a finite number of steps, after computing all its members. For instance, this holds if the subsemiring $\mathcal{L}^*(\delta^A, \sigma^A, \varphi)$ of $\mathcal{L}^*$ generated by all membership values taken by δ^A, σ^A and φ is finite (but not only in this case). If k denotes the number of elements of this subsemiring, then the collection $\{\varphi_u\}_{u \in X^*}$ can have at most k^n different members.

The tree that is constructed by this algorithm is a full m-ary tree. At the end of the algorithm, the tree can contain at most k^n internal vertices, and according to the well-known theorem on full m-ary trees, the total number of vertices is at most $mk^n + 1$. In the construction of any single vertice we can first perform a composition of the form $\varphi_u \circ \delta_x^A$, and then a composition of the form $(\varphi_u \circ \delta_x^A) \circ \varphi$, both of which have computation time $O(n^2(c_\otimes + c_\vee))$. Therefore,

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

the computation time for all performed compositions is $O(mn^2k^n(c_\otimes + c_\vee))$. Moreover, the tree T has at most mk^n edges, and the computation time of their forming is $O(mk^n)$.

The time-consuming part of the procedure is the check whether the just computed fuzzy set is a copy of some previously computed fuzzy set. After we have constructed the jth fuzzy set, for some $j \in \mathbb{N}$ such that $2 \leqslant j \leqslant mk^n + 1$, we compare it with the previously constructed fuzzy sets which correspond to non-closed vertices, whose number is at most $\min\{j-1, k^n\}$. Therefore, the total number of performed checks does not exceed $1 + 2 + \cdots + k^n + (m-1)k^n \cdot k^n = \frac{1}{2}k^n(k^n+1) + (m-1)k^{2n}$. As the computation time of any single check is $O(n)$, the computation time for all performed checks is $O(mnk^{2n})$. Summarizing all the above we conclude that the computation time of the whole algorithm is $O(mnk^{2n})$, the same as the computation time of the part in which for any newly-constructed fuzzy set we check whether it is a copy of some previously computed fuzzy set.

Note that the number k is characteristic of the fuzzy finite automaton $\mathcal{A}$, and it is not a general characteristic of the semiring $\mathcal{L}^*$ and its finitely generated subsemirings. However, if we consider fuzzy automata over a finite lattice, then we can assume that k is the number of elements of this lattice. Moreover, if $\mathcal{L}$ is the Gödel structure, then the set of all membership values taken by the fuzzy relations $\{\delta_x^A\}_{x\in X}$ and φ, and the fuzzy set σ^A is a subsemiring of $\mathcal{L}^*$, and the number of these values does not exceed $mn^2 + n$, so we can use that number instead of k.

In a similar way we can provide the following algorithm which constructs the automaton $\mathcal{A}^\psi$, for some fuzzy relation ψ on A, and analyze its computation time.

Algorithm 4.2 *(Construction of the automaton $\mathcal{A}^\psi$).* The input of this algorithm is a fuzzy finite automaton $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ with n states, over a finite alphabet X with m letters, and a fuzzy relation ψ on A, and the output is the crisp-deterministic fuzzy automaton $\mathcal{A}^\psi = (A^\psi, \psi^\varepsilon, \delta^\psi, \tau^\psi)$.

The procedure is to construct the *transition tree* of $\mathcal{A}^\psi$ directly from $\mathcal{A}$, and during this procedure we use pointers $s(\cdot)$ which points vertices of the tree under construction to the corresponding integers. The transition tree of $\mathcal{A}^\psi$ is constructed inductively as follows:

(A1) The root of the tree is $\psi^\varepsilon = \psi \circ \tau^A$, and we put $T_0 = \{\psi^\varepsilon\}$ and $s(\psi^\varepsilon) = 1$, and we compute the value $\tau^\psi(\psi^\varepsilon) = \sigma^A \circ \psi^\varepsilon$.

(A2) After the ith step let a tree T_i have been constructed, and vertices in T_i have been labeled either 'closed' or 'non-closed'. The meaning of these two terms will be made clear in the sequel.

(A3) In the next step we construct a tree T_{i+1} by enriching T_i in the following way: for any non-closed leaf ψ^u occurring in T_i, where $u \in X^*$, and each $x \in X$ we add a vertex $\psi^{xu} = \psi \circ \delta_x^A \circ \psi^u$ and an edge from ψ^u to ψ^{xu} labelled by x. Simultaneously, we check whether ψ^{xu} is a fuzzy set that has already been constructed. If it is true, if ψ^{xu} is equal to some previously computed ψ^v, we mark ψ^{xu} as closed and set $s(\psi^{xu}) = s(\psi^v)$. Otherwise, we compute the value $\tau^\psi(\psi^{xu}) = \sigma^A \circ \psi^{xu}$ and set $s(\psi^{xu})$ to be the next unassigned integer. The procedure terminates when all leaves are marked closed.

(A4) When the transition tree of $\mathcal{A}^\psi$ is constructed, we erase all closure marks and glue leaves to interior vertices with the same pointer value. The diagram that results is the transition graph of $\mathcal{A}^\psi$.

When ψ is the crisp equality on A, Algorithm 4.2 produces the reverse Nerode automaton $\mathcal{A}_{\overline{N}}$ of $\mathcal{A}$.

The conditions under which the above procedure terminates in a finite number of steps and its computation time can be analyzed analogously as in Algorithm 4.1. The only difference is that instead of the subsemiring $\mathcal{L}^*(\delta^A, \sigma^A, \varphi)$ here we consider the subsemiring $\mathcal{L}^*(\delta^A, \tau^A, \psi)$ of $\mathcal{L}^*$ generated by all membership values taken by δ^A, τ^A and ψ.

As we have said earlier, algorithms for computing the greatest right and left invariant fuzzy quasi-orders and the greatest weakly right and left invariant fuzzy quasi-orders on a fuzzy finite automaton were provided in [59]. Here we present these algorithms and we perform an analysis of their computation time.

Algorithm 4.3 *(Computation of the greatest right invariant fuzzy quasi-order).* The input of this algorithm is a fuzzy finite automaton $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ with n states, over a finite alphabet X with m letters. The algorithm computes the greatest right invariant fuzzy quasi-order φ^{ri} on $\mathcal{A}$.

The procedure constructs the sequence of fuzzy quasi-orders $\{\varphi_k\}_{k\in\mathbb{N}}$, in the following way:

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

(A1) In the first step we set $\varphi_1 = \tau^A/\tau^A$, where τ^A/τ^A is the right residual of τ^A by τ^A (cf. (9)).
(A2) After the kth step let φ_k be the fuzzy quasi-order that has been constructed.
(A3) In the next step we construct the fuzzy quasi-order φ_{k+1} by means of the formula

$$\varphi_{k+1} = \varphi_k \wedge \Big[\bigwedge_{x\in X} (\delta_x^A \circ \varphi_k)/(\delta_x^A \circ \varphi_k)\Big]. \tag{37}$$

(A4) Simultaneously, we check whether $\varphi_{k+1} = \varphi_k$.
(A5) When we find the smallest number s such that $\varphi_{s+1} = \varphi_s$, the procedure of constructing the sequence $\{\varphi_k\}_{k\in\mathbb{N}}$ terminates and $\varphi^{\mathrm{ri}} = \varphi_s$.

If the subalgebra $\mathcal{L}(\delta^A, \tau^A)$ of $\mathcal{L}$, generated by all membership values taken by δ^A and τ^A, satisfies DCC, the algorithm terminates in a finite number of steps.

Consider the computation time of this algorithm. In (A1) we compute τ^A/τ^A, which can be done in time $O(n^2 c_{\to})$. In (A3) we first compute all compositions $\delta_x^A \circ \varphi_k$, and if these computations are performed according to the definition of composition of fuzzy relations, their computation time is $O(mn^3(c_{\otimes} + c_{\vee}))$. Then we compute φ_{k+1} by means of (37), and the computation time of this part is $O(mn^3(c_{\to} + c_{\wedge}))$. Thus, the total computation time of (A3) is $O(mn^3(c_{\to} + c_{\wedge} + c_{\otimes} + c_{\vee}))$. In (A4), the computation time to check whether $\varphi_{k+1} = \varphi_k$ is $O(n^2)$.

The hardest problem is to estimate the number of steps, in the case when it is finite. Consider fuzzy relations φ_k as fuzzy matrices. After each step in the construction of the sequence $\{\varphi_k\}_{k\in\mathbb{N}}$ we check whether some entry has changed its value, and the algorithm terminates after the first step in which there was no change. Suppose that $\mathcal{L}(\delta^A, \tau^A)$ satisfies DCC. Then $\{\{\varphi_k(a,b)\}_{k\in\mathbb{N}} \mid (a,b) \in A^2\}$ is a finite collection of finite sequences, so there exists $s \in \mathbb{N}$ such that the number of different elements in each of these sequences is less than or equal to s. As the sequence $\{\varphi_k\}_{k\in\mathbb{N}}$ is descending, each entry can change its value at most $s-1$ times, and the total number of changes is less than or equal to $(s-1)(n^2-n)$ (the diagonal values must always be 1). Therefore, the algorithm terminates after at most $(s-1)(n^2-n)+2$ steps (in the first and last step values do not change).

Summing up, we get that the total computation time for the whole algorithm is $O(smn^5(c_{\to} + c_{\wedge} + c_{\otimes} + c_{\vee}))$, and hence, the algorithm is polynomial-time.

Let us note that the number s is characteristic of the sequence $\{\varphi_k\}_{k\in\mathbb{N}}$, and in general it is not characteristic of the algebra $\mathcal{L}(\delta^A, \tau^A)$. However, in some cases the number of different elements in all descending chains in $\mathcal{L}(\delta^A, \tau^A)$ may have an upper bound s. For example, if the algebra $\mathcal{L}(\delta^A, \tau^A)$ is finite, then we can assume that s is the number of elements of this algebra. In particular, if $\mathcal{L}$ is the Gödel structure, then the only values that can be taken by fuzzy relations $\{\varphi_k\}_{k\in\mathbb{N}}$ are 1 and those taken by δ^A and τ^A. In this case, if j is the number of all values taken by δ^A and τ^A, then the algorithm terminates after at most $j(n^2-n)+2$ steps, and total computation time is $O(jmn^5)$. Since $j \leqslant mn^2 + n^2$, total computation time can also be roughly expressed as $O(m^2n^7)$.

Algorithm 4.4 *(Computation of the greatest left invariant fuzzy quasi-order).* The input of this algorithm is a fuzzy finite automaton $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ with n states, over a finite alphabet X with m letters. The algorithm computes the greatest left invariant fuzzy quasi-order ψ^{li} on $\mathcal{A}$.

The procedure constructs the sequence of fuzzy quasi-orders $\{\psi_k\}_{k\in\mathbb{N}}$, in the following way:

(A1) In the first step we set $\psi_1 = \sigma^A \backslash \sigma^A$.
(A2) After the kth step let ψ_k be the fuzzy quasi-order that has been constructed.
(A3) In the next step we construct the fuzzy quasi-order ψ_{k+1} by means of the formula

$$\psi_{k+1} = \psi_k \wedge \Big[\bigwedge_{x\in X} (\psi_k \circ \delta_x^A)\backslash(\psi_k \circ \delta_x^A)\Big].$$

(A4) Simultaneously, we check whether $\psi_{k+1} = \psi_k$.
(A5) When we find the smallest number s such that $\psi_{s+1} = \psi_s$, the procedure of constructing the sequence $\{\psi_k\}_{k\in\mathbb{N}}$ terminates and $\psi^{\mathrm{li}} = \psi_s$.

The conditions under which this procedure terminates in a finite number of steps and its computation time can be analyzed analogously as in Algorithm 4.3.

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

Algorithm 4.5 *(Computation of the greatest weakly right invariant fuzzy quasi-order).* The input of this algorithm is a fuzzy finite automaton $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ with n states, over a finite alphabet X with m letters. The algorithm computes the greatest weakly right invariant fuzzy quasi-order φ^{wri} on $\mathcal{A}$.

The procedure consists of two parts:

(A1) First we compute all members of the family $\{\tau_u^A \mid u \in X^*\}$, using Algorithm 4.2.
(A2) Then we compute φ^{wri} by means of formula

$$\varphi^{\mathrm{wri}} = \bigwedge_{u \in X^*} \tau_u^A / \tau_u^A.$$

Clearly, this procedure terminates in a finite number of steps under the same conditions as Algorithm 4.2. Under these conditions the computation time of the part (A1) is $O(mnk^{2n})$, and since it dominates over the computation time of (A2), which is $O(k^n c_\wedge + n^2 c_\to)$, we conclude that the computation time of the whole algorithm is $O(mnk^{2n})$, the same as for Algorithms 4.1 and 4.2.

Analogous analysis can be performed for the following algorithm that computes the greatest weakly left invariant fuzzy quasi-order on a fuzzy automaton.

Algorithm 4.6 *(Computation of the greatest weakly left invariant fuzzy quasi-order).* The input of this algorithm is a fuzzy finite automaton $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ over a finite alphabet X. The algorithm computes the greatest weakly left invariant fuzzy quasi-order ψ^{wli} on $\mathcal{A}$.

The procedure consists of two parts:

(A1) First we compute all members of the family $\{\sigma_u^A \mid u \in X^*\}$, using Algorithm 4.1.
(A2) Then we compute ψ^{wli} by means of formula

$$\psi^{\mathrm{wli}} = \bigwedge_{u \in X^*} \sigma_u^A \backslash \sigma_u^A.$$

Finally, we turn to the construction of the children automaton $\mathcal{A}_\varphi^c$.

Algorithm 4.7 *(Construction of the children automaton $\mathcal{A}_\varphi^c$).* The input of this algorithm is a fuzzy finite automaton $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ with n states, over a finite alphabet $X = \{x_1, \ldots, x_m\}$ with m letters, and the output is the children automaton $\mathcal{A}_\varphi^c = (A_\varphi^c, \varphi_\varepsilon^c, \delta_\varphi^c, \tau_\varphi^c)$, where φ is the greatest weakly right invariant fuzzy quasi-order or the greatest right invariant fuzzy quasi-order on $\mathcal{A}$.

The procedure is to construct simultaneously the *transition tree* of $\mathcal{A}_\varphi$ and the *transition graph* of $\mathcal{A}_\varphi^c$ directly from $\mathcal{A}$. Except the pointer $s(\cdot)$ used in Algorithm 4.1, we also use another pointer $t(\cdot)$ which points vertices of the transition graph under construction to the corresponding integers. The transition tree of $\mathcal{A}_\varphi$ and the transition graph of $\mathcal{A}_\varphi^c$ are constructed in the following way:

(A1) We compute φ using one of Algorithms 4.5 and 4.3, and construct the transition tree T of $\mathcal{A}_\varphi$ using Algorithm 4.1.
(A2) To each non-closed vertex φ_u of the tree T we assign a vertex φ_u^c of a graph G as follows: When all children $\varphi_{ux_1}, \ldots, \varphi_{ux_m}$ of φ_u in the tree T are formed, we form the vertex $\varphi_u^c = (\varphi_{ux_1}, \ldots, \varphi_{ux_m}, \varphi_u \circ \tau^A)$ in the graph G. Simultaneously, we check whether φ_u^c is an $(m+1)$-tuple that has already been constructed. If it is true, if φ_u^c is equal to some previously computed φ_v^c, we mark φ_u^c as closed and set $t(\varphi_u^c) = t(\varphi_v^c)$. Otherwise, we put $\tau_\varphi^c(\varphi_u^c) = \varphi_u \circ \tau^A$ and set $t(\varphi_u^c)$ to be the next integer that has not been used as a value for $t(\cdot)$.
(A3) For each non-closed vertex φ_u^c of the graph G and each $x \in X$, if φ_v is a non-closed vertex in T such that $s(\varphi_{ux}) = s(\varphi_v)$, in the graph G we add an edge from φ_u^c to φ_v^c labeled by x.
(A4) When the graph G is constructed, we glue closed vertices to non-closed vertices with the same pointer value, and erase closure marks. The diagram that results is the transition graph of $\mathcal{A}_\varphi^c$.

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

According to Theorems 3.2 and 3.12, $\mathcal{A}_\varphi$ is isomorphic to $\mathcal{B}_N$, where $\mathcal{B} = \mathcal{A}/\varphi$ is the afterset fuzzy automaton of $\mathcal{A}$ with respect to φ, and $\mathcal{A}_\varphi^c$ is isomorphic to $\mathcal{B}_N^c$, and by Theorem 3.7 [33], $\mathcal{B}_N^c$ is finite if and only if $\mathcal{B}_N$ is finite. Therefore, $\mathcal{A}_\varphi^c$ is finite if and only if $\mathcal{A}_\varphi$ is finite.

As in the analysis of Algorithm 4.1, consider the case when the subalgebra $\mathcal{L}(\delta^A, \sigma^A, \tau^A)$ of $\mathcal{L}$ is finite and has l elements. As we have already seen, the computation time of the part (A1) is $O(mnl^{2n})$.

When in (A2) we construct a vertex of the graph G, as an $(m+1)$-tuple, all its components have already been computed during construction of the tree T, and all that remains to do is to point to them (or to their addresses). Hence, the computation time of forming every single vertex of the graph G is $O(m)$, and since there are at most l^n vertices in G, the computation time of forming all vertices of G is $O(ml^n)$. During construction of the tree T we also found which vertices in T are equal as fuzzy sets, and they received the same pointer values. Therefore, when we check equality of two $(m+1)$-tuples in A_φ^c we only need to check the equality of pointer values assigned to their components, and the computation time of such checking is $O(m)$. As the total number of checks performed in (A2) does not exceed $1+2+\cdots+(l^n-1) = \frac{1}{2}(l^n-1)l^n$, we have that computation time of all checks performed in (A2) is $O(ml^{2n})$. Finally, in (A3) we form at most ml^n edges in the graph G, and the computation time of this part is $O(ml^n)$.

Therefore, the most expensive part of this algorithm is (A1), and the computation time of the whole algorithm is $O(mnl^{2n})$, the same as for (A1), i.e., the same as for Algorithm 4.7.

Finally we give a remark regarding the computation time of the Brzozowski type algorithm for fuzzy finite automata. Let $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ be a fuzzy finite automaton with n states and m input letters, and suppose that the subsemiring $\mathcal{L}^*(\delta^A, \sigma^A, \tau^A)$ of the semiring $\mathcal{L}^*$, generated by all membership values taken by δ^A, σ^A and τ^A, is finite and has k elements. The first round of the application of the Brzozowski type procedure to $\mathcal{A}$ produces the reverse Nerode automaton of $\mathcal{A}$ having at most k^n states, and the computation time of this round is $O(mnk^{2n})$. The second round may start from an exponentially larger automaton, but despite that, this round produces a minimal crisp-deterministic fuzzy automaton equivalent to $\mathcal{A}$, an automaton that is not greater than the Nerode automaton of $\mathcal{A}$, which can not have more than k^n states. Thus, the resulting transition tree can not have more than k^n internal vertices, and the total number of vertices is not greater than $mk^n + 1$. However, computation of any single vertex may be considerably more expensive than in previous algorithms because here we multiply vectors of size r and matrices of size $r \times r$, where $r \leqslant k^n$. Therefore, computation of any single vertex of the transition tree in the second round requires time $O(k^{2n}(c_\otimes + c_\vee))$, and the computation time for all vertices is $O(mk^{3n}(c_\otimes + c_\vee))$. Since the tree has at most mk^n edges, the computation time of their forming is $O(mk^n)$.

When for any newly-constructed fuzzy set we check whether it is a copy of some previously computed fuzzy set, the total number of performed checks is $\frac{1}{2}k^n(k^n+1) + (m-1)k^{2n}$, the same as in previous algorithms, but here any single check has the computation time $O(k^n)$, so the computation time for all performed checks is $O(mk^{3n})$. Hence, the computation time of the whole algorithm is $O(mk^{3n}(c_\otimes + c_\vee))$, or $O(mk^{3n})$, if the operations $\otimes$ and $\vee$ can be performed in constant time. Accordingly, the Brzozowski type algorithm is somewhat slower than the other algorithms discussed here, but its performances can be improved if instead of the construction of the reverse Nerode automaton we use the construction of the automaton corresponding to the greatest right invariant or weakly right invariant fuzzy quasi-order on $\mathcal{A}$, or the construction of its children automaton.

Now we provide several illustrative computational examples.

Example 4.8. Let $\mathcal{A}$ be a Boolean automaton over the two-element alphabet $X = \{x, y\}$ given by the transition graph shown in Fig. 1.

The transition tree and the transition graph of the Nerode automaton $\mathcal{A}_N$ of $\mathcal{A}$, constructed by means of Algorithm 4.1, are presented in Fig. 2. We see that the Nerode automaton $\mathcal{A}_N$ has 7 states.

By means of Algorithms 4.3 and 4.5 we compute the greatest right invariant fuzzy quasi-order φ^{ri} and the greatest weakly right invariant fuzzy quasi-order φ^{wri} on $\mathcal{A}$, which are represented by the following Boolean matrices:

$$\varphi^{\mathrm{ri}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}, \qquad \varphi^{\mathrm{wri}} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

Then, using Algorithm 4.1 we construct automata $\mathcal{A}_{\varphi^{\mathrm{ri}}}$ and $\mathcal{A}_{\varphi^{\mathrm{wri}}}$, whose transition trees and graphs are presented in Figs. 3 and 4, respectively. The automaton $\mathcal{A}_{\varphi^{\mathrm{ri}}}$ has 5 states, whereas the automaton $\mathcal{A}_{\varphi^{\mathrm{wri}}}$ has 3 states. Finally, using

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

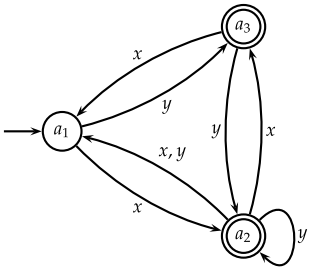

Fig. 1. The transition graph of the fuzzy automaton $\mathcal{A}$ from Example 4.8.

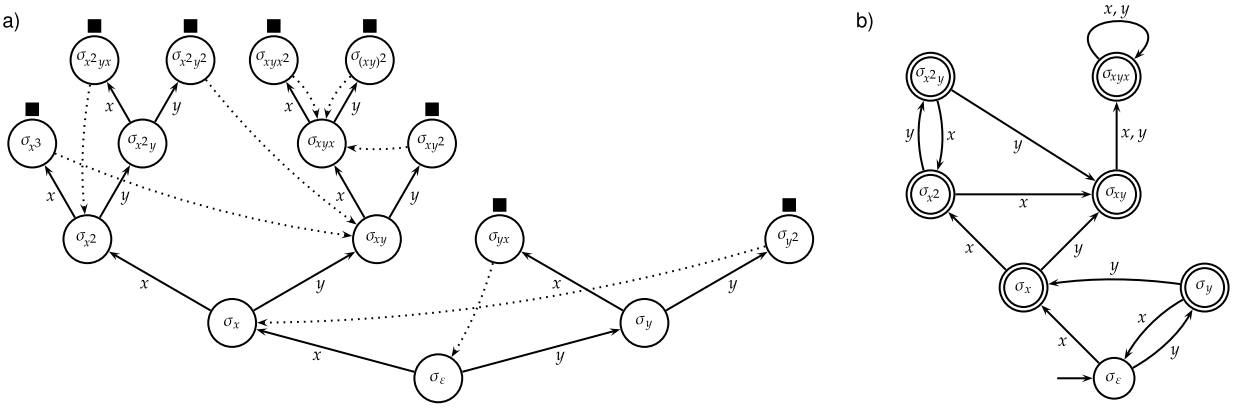

Fig. 2. The transition tree (a) and the transition graph (b) of the Nerode automaton $\mathcal{A}_N$ of the fuzzy automaton $\mathcal{A}$ from Example 4.8.

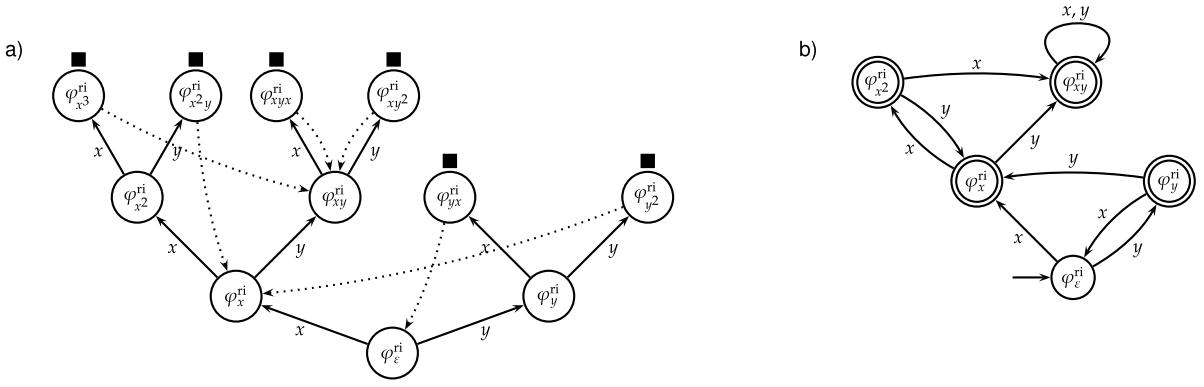

Fig. 3. The transition tree (a) and the transition graph (b) of the automaton $\mathcal{A}_{\varphi^{\mathrm{ri}}}$ for the fuzzy automaton $\mathcal{A}$ from Example 4.8.

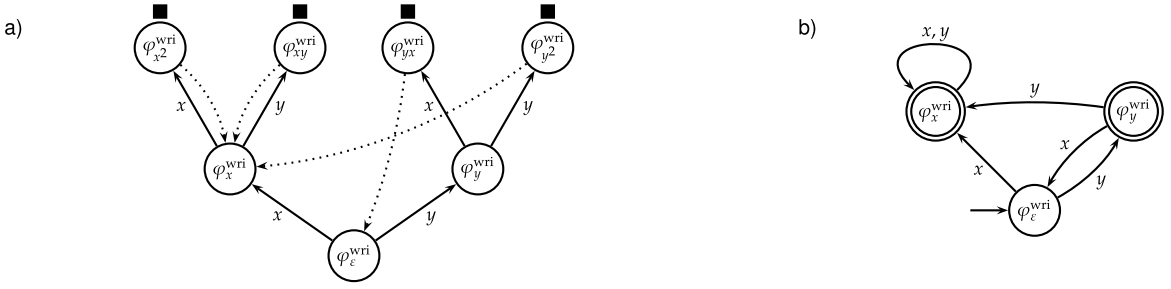

Fig. 4. The transition tree (a) and the transition graph (b) of the automaton $\mathcal{A}_{\varphi^{\mathrm{wri}}}$ for the fuzzy automaton $\mathcal{A}$ from Example 4.8.

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

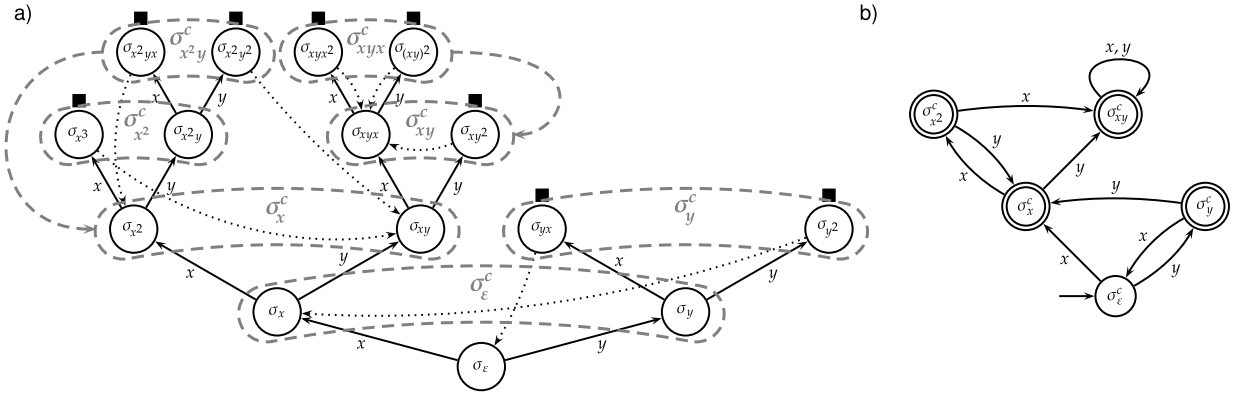

Fig. 5. Construction of the transition graph of the children automaton $\mathcal{A}_N^c$ from the transition tree of the Nerode automaton $\mathcal{A}_N$ from Fig. 2.

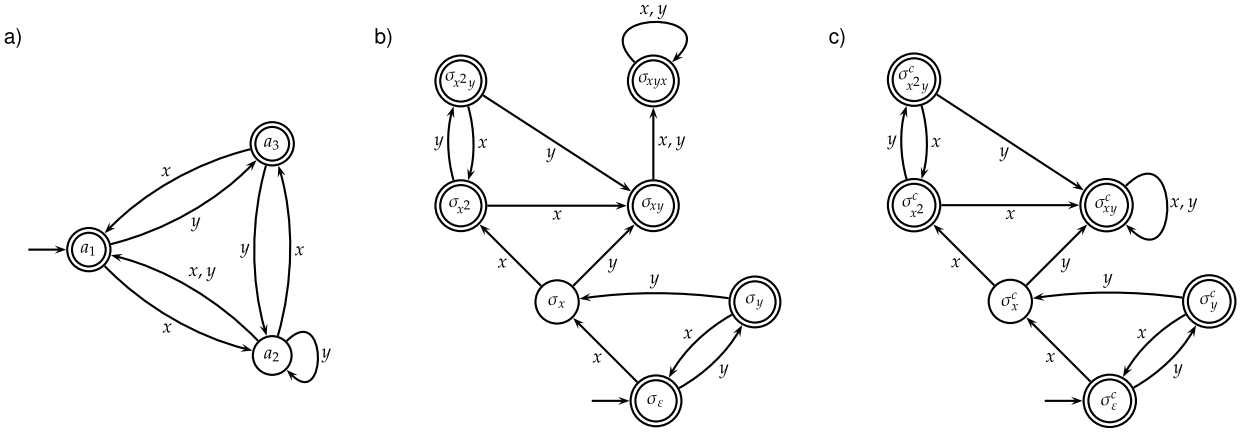

Fig. 6. Transition graphs of the fuzzy automaton $\mathcal{A}$ from Example 4.9 (a), its Nerode automaton $\mathcal{A}_N$ (b), and the children automaton $\mathcal{A}_N^c$ (c).

Algorithm 4.7 and the transition tree of $\mathcal{A}_N$ from Fig. 2, we construct the children auto maton $\mathcal{A}_N^c$ of $\mathcal{A}_N$ presented in Fig. 5. Clearly, $\mathcal{A}_N^c$ is isomorphic to $\mathcal{A}_{\varphi^{\mathrm{ri}}}$.

According to Theorem 3.4, the number of states of the automaton $\mathcal{A}_{\varphi^{\mathrm{ri}}}$ is less than or equal to the number of states of the Nerode automaton $\mathcal{A}_N$, for every fuzzy automaton $\mathcal{A}$. This example shows that $\mathcal{A}_{\varphi^{\mathrm{ri}}}$ can be strictly smaller than $\mathcal{A}_N$. The example also shows that the greatest weakly right invariant fuzzy quasi-order φ^{wri} can give better results in determinization than the greatest right invariant fuzzy quasi-order φ^{ri}.

Finally, we find that all aftersets of fuzzy quasi-orders φ^{ri} and φ^{wri} are different, which means that φ^{ri} and φ^{wri} do not reduce the number of states of the automaton $\mathcal{A}$, but despite this, they produce crisp-deterministic fuzzy automata which are smaller than the Nerode automaton of $\mathcal{A}$.

Example 4.9. Let $\mathcal{A}$ be a Boolean automaton over the two-element alphabet $X = \{x, y\}$ given by the transition graph shown in Fig. 6 a). The transition graphs of the Nerode automaton $\mathcal{A}_N$ and its children automaton $\mathcal{A}_N^c$, constructed by means of Algorithms 4.1 and 4.7, are represented by Fig. 6 b) and c). Clearly, the Nerode automaton $\mathcal{A}_N$ has 7 states, and its children automaton $\mathcal{A}_N^c$ has 6 states.

On the other hand, both the greatest right invariant fuzzy quasi-order φ^{ri} and the greatest weakly right invariant fuzzy quasi-order φ^{wri} on $\mathcal{A}$, computed using Algorithms 4.3 and 4.5, are equal to the equality relation on the set of states of $\mathcal{A}$, so both automata $\mathcal{A}_{\varphi^{\mathrm{ri}}}$ and $\mathcal{A}_{\varphi^{\mathrm{wri}}}$ are isomorphic to $\mathcal{A}_N$. Therefore, we have that $|\mathcal{A}_N^c| < |\mathcal{A}_N| = |\mathcal{A}_{\varphi^{\mathrm{ri}}}| = |\mathcal{A}_{\varphi^{\mathrm{wri}}}|$, which demonstrates that construction of the children auto maton, applied to the Nerode automaton of $\mathcal{A}$, can give better results in determinization of $\mathcal{A}$ than constructions based on the greatest right invariant and weakly right invariant fuzzy quasi-orders on $\mathcal{A}$.

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

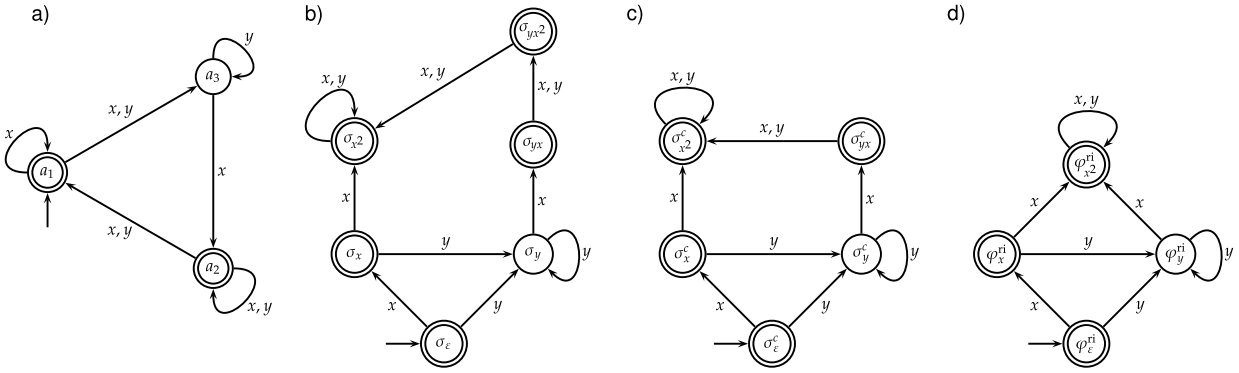

Fig. 7. Transition graphs of the fuzzy automaton $\mathcal{A}$ from Example 4.10 (a), its Nerode automaton $\mathcal{A}_N$ (b), the children automaton $\mathcal{A}_N^c$ of $\mathcal{A}_N$ (c), and the automaton $\mathcal{A}_{\varphi^{\mathrm{ri}}} \cong \mathcal{A}_{\varphi^{\mathrm{wri}}}$ (d).

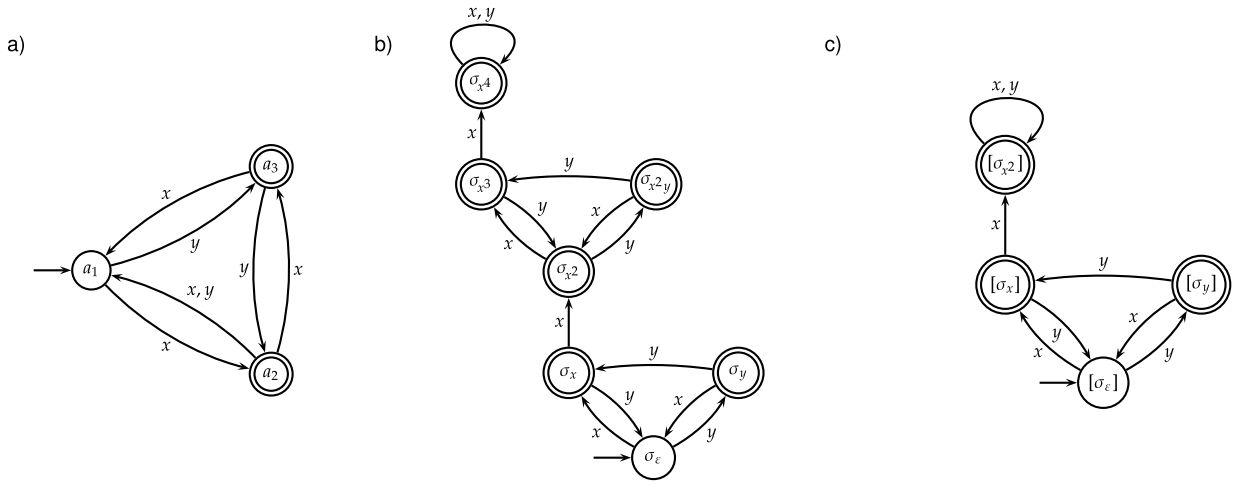

Fig. 8. Transition graphs of the fuzzy automaton $\mathcal{A}$ from Example 4.11 (a), its Nerode automaton $\mathcal{A}_N$ (b), and the minimal crisp-deterministic fuzzy automaton equivalent to $\mathcal{A}_N$ (c).

Example 4.10. Let $\mathcal{A}$ be a Boolean automaton over the two-element alphabet $X = \{x, y\}$ given by the transition graph shown in Fig. 7 a). The Nerode automaton $\mathcal{A}_N$ and the children automaton $\mathcal{A}_N^c$ are represented by graphs in Fig. 7 b) and c). Moreover, using Algorithms 4.3 and 4.5 we obtain that

$$\varphi^{\mathrm{ri}} = \varphi^{\mathrm{wri}} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix},$$

so automata $\mathcal{A}_{\varphi^{\mathrm{ri}}}$ and $\mathcal{A}_{\varphi^{\mathrm{wri}}}$ are mutually isomorphic and are given by the transition graph in Fig. 7d), and we conclude that $|\mathcal{A}_{\varphi^{\mathrm{ri}}}| = |\mathcal{A}_{\varphi^{\mathrm{wri}}}| < |\mathcal{A}_N^c| < |\mathcal{A}_N|$.

Thus, in contrast to the previous one, this example shows that there are cases where determinization of a fuzzy automaton $\mathcal{A}$ by means of the greatest right invariant and weakly right invariant fuzzy quasi-orders can give better results than construction of the children automaton of the Nerode automaton of $\mathcal{A}$.

Example 4.11. Let $\mathcal{A}$ be a Boolean automaton over the two-element alphabet $X = \{x, y\}$ given by the transition graph shown in Fig. 8 a). When we compute φ^{ri} and φ^{wri} we obtain that both of them are equal to the equality relation on the set of states of $\mathcal{A}$, and therefore, both automata $\mathcal{A}_{\varphi^{\mathrm{ri}}}$ and $\mathcal{A}_{\varphi^{\mathrm{wri}}}$ are isomorphic to the Nerode automaton $\mathcal{A}_N$, which is represented by the transition graph in Fig. 8 b). Moreover, we obtain that the children automaton $\mathcal{A}_N^c$ of $\mathcal{A}_N$ is also isomorphic to the Nerode automaton $\mathcal{A}_N$. Hence, in this case none of the methods discussed in this paper does not

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

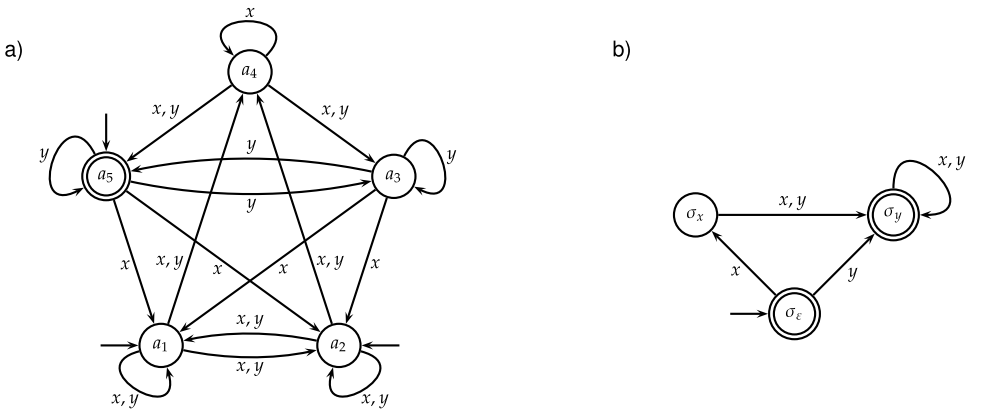

Fig. 9. The transition graph of the fuzzy automaton from Example 4.12 (a), and its Nerode automaton $\mathcal{A}_N$ (b).

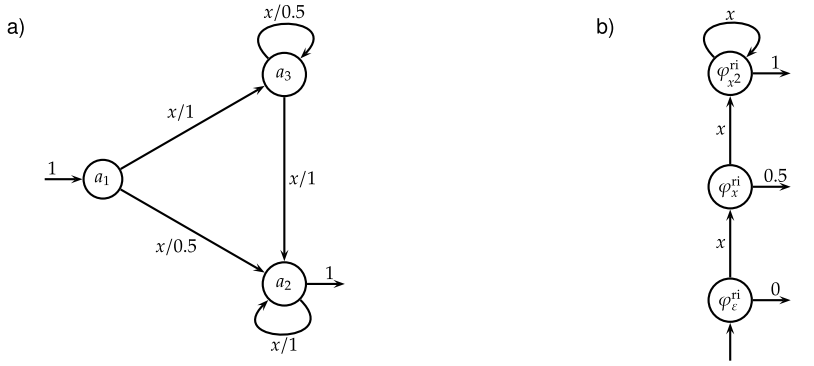

Fig. 10. Transition graphs of the fuzzy automaton $\mathcal{A}$ from Example 4.13 (a), and the automaton $\mathcal{A}_{\varphi^{\mathrm{ri}}}$ (b).

give an automaton with smaller number of states than the Nerode automaton $\mathcal{A}_N$. It should be noted that the Nerode automaton $\mathcal{A}_N$ is not minimal, the minimal deterministic automaton equivalent to $\mathcal{A}_N$ is represented by the graph shown in Fig. 8 c).

Example 4.12. Let $\mathcal{A}$ be a Boolean automaton over the two-element alphabet $X = \{x, y\}$ given by the transition graph shown in Fig. 9 a). The transition graph of the Nerode automaton $\mathcal{A}_N$ of $\mathcal{A}$ is given in Fig. 9 b). Using Algorithms 4.3 and 4.5 we obtain that

$$\varphi^{\mathrm{ri}} = \varphi^{\mathrm{wri}} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix},$$

and we can easily show that $\mathcal{A}_{\varphi^{\mathrm{ri}}}$ is isomorphic to the Nerode automaton $\mathcal{A}_N$. Note that φ^{ri} has 4 different aftersets, which means that the afterset fuzzy automaton of $\mathcal{A}$ with respect to φ^{ri} has 4 states. Therefore, although φ^{ri} reduces the number of states of $\mathcal{A}$, it does not give an automaton with smaller number of states than the Nerode automaton $\mathcal{A}_N$.

Example 4.13. Let $\mathcal{A}$ be an automaton over the one-element alphabet $X = \{x\}$ and the Goguen (product) structure given by the transition graph shown in Fig. 10 a).

It is easy to check that $\sigma_\varepsilon = [1\ 0\ 0]$, $\sigma_x = [0\ 0.5\ 1]$, and $\sigma_{x^n} = [0\ 1\ 0.5^{n-1}]$, for each $n \in \mathbb{N}$, $n \geqslant 2$. Therefore, the Nerode automaton of $\mathcal{A}$ has infinitely many states.

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

On the other hand, using Algorithms 4.3 and 4.5 we obtain that

$$\varphi^{\mathrm{ri}} = \varphi^{\mathrm{wri}} = \begin{bmatrix} 1 & 0 & 0.5 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix},$$

and we construct the automaton $\mathcal{A}_{\varphi^{\mathrm{ri}}}$ which is shown in Fig. 10 b).

Therefore, although the Nerode automaton of $\mathcal{A}$ is infinite, using the greatest right invariant fuzzy quasi-order on $\mathcal{A}$ we obtain a finite crisp-deterministic fuzzy automaton which is equivalent to $\mathcal{A}$.

5. Concluding remarks

Fuzzy automata originated in the late 1960s as a model of computation that combines the capabilities of classical automata and fuzzy logic and is able to handle the uncertainty which is inherent in many applications. Over the years they have gained significant applications in many areas such as approximate string matching and searching, neural networks and neuro-fuzzy systems, fuzzy signal processing, and clinical monitoring, to name but a few [5,22,47,55, 58,61,65]. In recent years, fuzzy automata were also used as the basic model for fuzzy discrete-event systems, which have been successfully applied to biomedical control for HIV/AIDS treatment planning, robotic control, intelligent vehicle control, waste-water treatment, examination of chemical reactions, fault diagnosis, and in other fields [6–8,14, 21,39–43,54,56,59,68].

The most general and most widely studied type of fuzzy automata, with fuzzy sets of initial and terminal states and transitions that form fuzzy relations, originated as an immediate generalization of classical nondeterministic automata. They are called simply fuzzy automata. A special case of these automata are fuzzy automata with a single crisp initial state and a deterministic transition function, where the fuzziness is entirely concentrated in the fuzzy set of terminal states. They were first introduced by Bělohlávek in [3], and we call them crisp-deterministic fuzzy automata. The third type of fuzzy automata, used in all articles dealing with fuzzy discrete-event systems (except the articles [43,59]), are automata whose set of states are fuzzy subsets of a given set and the transition function acts deterministically on these fuzzy sets. In [60] they were called automata with fuzzy states.

Each fuzzy automaton can be transformed into an equivalent crisp-deterministic fuzzy automaton. This has been proven by Bělohlávek [3] for fuzzy automata over a complete lattice, and Li and Pedrycz [38] have given the same result for fuzzy automata over a lattice-ordered monoid. However, the resulting crisp-deterministic fuzzy automaton can have an exponentially larger number of states than the original fuzzy automaton, and for fuzzy automata over some structures of membership values it may even be infinite. For this reason it is extremely important to develop determinization methods that will give as small as possible crisp-deterministic fuzzy automata. Two such methods have been developed in [25,33], while in [34,45] two determinization methods that result in minimal crisp-deterministic fuzzy automata have been provided, which are called canonization methods. This paper has provided further improvements of these methods. We have developed methods that give smaller automata than all previously mentioned methods, and methods that can be used in the first stage of the aforementioned canonization methods in order to mitigate a combinatorial blow up of the number of states that may happen in this stage.

It is worth noting that all crisp-deterministic fuzzy automata constructed in this paper and in [25,33,45] are actually automata with fuzzy states. Moreover, it has been shown in [60] that each crisp-deterministic fuzzy automaton can be transformed into an equivalent automaton with fuzzy states, while every automaton with fuzzy states can be naturally viewed as a crisp-deterministic fuzzy automaton. Therefore, it is very important to perform a deeper analysis of the connections between fuzzy automata, crisp-deterministic fuzzy automata and automata with fuzzy states, especially from the aspect of their application in fuzzy discrete-event systems, what will be the subject of our future research. Let us also note that any fuzzy state can be regarded as a possibility distribution expressing that some states are more plausible than others. Such a way of viewing is basic in what is called partially observable Markov decision processes (cf., e.g., [11–13]), whose fuzzy counterparts have been discussed, for example, in [57,23]. Something similar to the idea of determinization is also present (as the subset construction) in work with POMDPs, where a POMDP should be converted into a perfect-observation Markov decision process. As the main problem here is that the resulting perfect-observation MDP is exponentially larger than the original POMDP, we find that it might be very useful to adapt and apply our determinization methods to POMDP, which we also intend to deal with in the future.

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

Acknowledgements

The authors express their gratitude to the reviewers for their helpful comments and suggestions, and to the area editor who gave very interesting ideas for further research.

References

[1] N.C. Basak, A. Gupta, On quotient machines of a fuzzy automaton and the minimal machine, Fuzzy Sets Syst. 125 (2002) 223–229.
[2] R. Bělohlávek, Fuzzy Relational Systems: Foundations and Principles, Kluwer, New York, 2002.
[3] R. Bělohlávek, Determinism and fuzzy automata, Inf. Sci. 143 (2002) 205–209.
[4] R. Bělohlávek, V. Vychodil, Fuzzy Equational Logic, Studies in Fuzziness and Soft Computing, Springer, Berlin, Heidelberg, 2005.
[5] A. Blanco, M. Delgado, M.C. Pegalajar, Fuzzy automaton induction using neural network, Int. J. Approx. Reason. 27 (2001) 1–26.
[6] Y.Z. Cao, M.S. Ying, Supervisory control of fuzzy discrete event systems, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 35 (2005) 366–371.
[7] Y.Z. Cao, M.S. Ying, Observability and decentralized control of fuzzy discrete-event systems, IEEE Trans. Fuzzy Syst. 14 (2006) 202–216.
[8] Y.Z. Cao, M.S. Ying, G.Q. Chen, State-based control of fuzzy discrete-event systems, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 37 (2007) 410–424.
[9] C. Câmpeanu, N. Sântean, S. Yu, Mergible states in large NFA, Theor. Comput. Sci. 330 (2005) 23–34.
[10] J.-M. Champarnaud, F. Coulon, NFA reduction algorithms by means of regular inequalities, Theor. Comput. Sci. 327 (2004) 241–253.
[11] K. Chatterjee, M. Chmelík, POMDPs under probabilistic semantics, Artif. Intell. 221 (2015) 46–72.
[12] K. Chatterjee, L. Doyen, T.A. Henzinger, Qualitative analysis of partially-observable Markov decision processes, in: P. Hliněný, A. Kučera (Eds.), MFCS 2010, in: Lecture Notes in Computer Science, vol. 6281, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 258–269.
[13] K. Chatterjee, L. Doyen, T.A. Henzinger, J.-F. Raskin, Algorithms for omega-regular games with imperfect information, Log. Methods Comput. Sci. 3 (3–4) (2007) 1–23.
[14] X. Chen, H. Xing, Nonblocking check in fuzzy discrete event systems based on observation equivalence, Fuzzy Sets Syst. 269 (2015) 47–64.
[15] W. Cheng, Z. Mo, Minimization algorithm of fuzzy finite automata, Fuzzy Sets Syst. 141 (2004) 439–448.
[16] M. Ćirić, M. Droste, J. Ignjatović, H. Vogler, Determinization of weighted finite automata over strong bimonoids, Inf. Sci. 180 (2010) 3497–3520.
[17] M. Ćirić, J. Ignjatović, M. Bašić, I. Jančić, Nondeterministic automata: equivalence, bisimulations, and uniform relations, Inf. Sci. 261 (2014) 185–218.
[18] M. Ćirić, J. Ignjatović, N. Damljanović, M. Bašić, Bisimulations for fuzzy automata, Fuzzy Sets Syst. 186 (2012) 100–139.
[19] M. Ćirić, J. Ignjatović, I. Jančić, N. Damljanović, Computation of the greatest simulations and bisimulations between fuzzy automata, Fuzzy Sets Syst. 208 (2012) 22–42.
[20] M. Ćirić, A. Stamenković, J. Ignjatović, T. Petković, Fuzzy relation equations and reduction of fuzzy automata, J. Comput. Syst. Sci. 76 (2010) 609–633.
[21] W. Deng, D. Qiu, Supervisory control of fuzzy discrete-event systems for simulation equivalence, IEEE Trans. Fuzzy Syst. 23 (1) (2015) 178–191.
[22] M. Doostfatemeh, S.C. Kremer, New directions in fuzzy automata, Int. J. Approx. Reason. 38 (2005) 175–214.
[23] N. Drougard, F. Teichteil-Königsbuch, J.-L. Farges, D. Dubois, Qualitative possibilistic mixed-observable MDPs, in: Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence, UAI 2013, 2013, pp. 192–201.
[24] J. Ignjatović, M. Ćirić, Formal power series and regular operations on fuzzy languages, Inf. Sci. 180 (2010) 1104–1120.
[25] J. Ignjatović, M. Ćirić, S. Bogdanović, Determinization of fuzzy automata with membership values in complete residuated lattices, Inf. Sci. 178 (2008) 164–180.
[26] J. Ignjatović, M. Ćirić, S. Bogdanović, Fuzzy homomorphisms of algebras, Fuzzy Sets Syst. 160 (2009) 2345–2365.
[27] J. Ignjatović, M. Ćirić, S. Bogdanović, T. Petković, Myhill–Nerode type theory for fuzzy languages and automata, Fuzzy Sets Syst. 161 (2010) 1288–1324.
[28] L. Ilie, S. Yu, Algorithms for computing small NFAs, in: MFCS 2002, in: Lecture Notes in Computer Science, vol. 2420, Springer-Verlag, Berlin, Heidelberg, 2002, pp. 328–340.
[29] L. Ilie, S. Yu, Reducing NFAs by invariant equivalences, Theor. Comput. Sci. 306 (2003) 373–390.
[30] L. Ilie, G. Navarro, S. Yu, On NFA reductions, in: J. Karhumäki, et al. (Eds.), Theory Is Forever, in: Lecture Notes in Computer Science, vol. 3113, Springer-Verlag, Berlin, Heidelberg, 2004, pp. 112–124.
[31] L. Ilie, R. Solis-Oba, S. Yu, Reducing the size of NFAs by using equivalences and preorders, in: A. Apostolico, M. Crochemore, K. Park (Eds.), CPM 2005, in: Lecture Notes in Computer Science, vol. 3537, Springer-Verlag, Berlin, Heidelberg, 2005, pp. 310–321.
[32] I. Jančić, Weak bisimulations for fuzzy automata, Fuzzy Sets Syst. 249 (2014) 49–72.
[33] Z. Jančić, J. Ignjatović, M. Ćirić, An improved algorithm for determinization of weighted and fuzzy automata, Inf. Sci. 181 (2011) 1358–1368.
[34] Z. Jančić, M. Ćirić, Brzozowski type determinization for fuzzy automata, Fuzzy Sets Syst. 249 (2014) 73–82.
[35] T. Jiang, B. Ravikumar, Minimal NFA problems are hard, SIAM J. Comput. 22 (6) (1993) 1117–1141.
[36] H. Lei, Y.M. Li, Minimization of states in automata theory based on finite lattice-ordered monoids, Inf. Sci. 177 (2007) 1413–1421.
[37] L. Li, D.W. Qiu, On the state minimization of fuzzy automata, IEEE Trans. Fuzzy Syst. 23 (2) (2015) 434–443.
[38] Y.M. Li, W. Pedrycz, Fuzzy finite automata and fuzzy regular expressions with membership values in lattice ordered monoids, Fuzzy Sets Syst. 156 (2005) 68–92.
[39] F. Lin, H. Ying, Modeling and control of fuzzy discrete event systems, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 32 (2002) 408–415.

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019

[40] F. Lin, H. Ying, R.D. MacArthur, J.A. Cohn, D. Barth-Jones, L.R. Crane, Decision making in fuzzy discrete event systems, Inf. Sci. 177 (2007) 3749–3763.
[41] F. Liu, D.W. Qiu, Decentralized supervisory control of fuzzy discrete event systems, Eur. J. Control 3 (2008) 234–243.
[42] F. Liu, D.W. Qiu, Diagnosability of fuzzy discrete event systems: a fuzzy approach, IEEE Trans. Fuzzy Syst. 17 (2) (2009) 372–384.
[43] J.P. Liu, Y.M. Li, The relationship of controllability between classical and fuzzy discrete-event systems, Inf. Sci. 178 (2008) 4142–4151.
[44] D.S. Malik, J.N. Mordeson, M.K. Sen, Minimization of fuzzy finite automata, Inf. Sci. 113 (1999) 323–330.
[45] I. Micić, Z. Jančić, J. Ignjatović, M. Ćirić, Determinization of fuzzy automata by means of the degrees of language inclusion, IEEE Trans. Fuzzy Syst. 23 (6) (2015) 2144–2153.
[46] J.N. Mordeson, D.S. Malik, Fuzzy Automata and Languages: Theory and Applications, Chapman & Hall/CRC, Boca Raton, London, 2002.
[47] C.W. Omlin, K.K. Thornber, C.L. Giles, Fuzzy finite state automata can be deterministically encoded into recurrent neural networks, IEEE Trans. Fuzzy Syst. 6 (1998) 76–89.
[48] K. Peeva, Finite L-fuzzy machines, Fuzzy Sets Syst. 141 (2004) 415–437.
[49] K. Peeva, Y. Kyosev, Fuzzy Relational Calculus: Theory, Applications, and Software (with CD-ROM), Advances in Fuzzy Systems – Applications and Theory, vol. 22, World Scientific, 2004.
[50] K. Peeva, Z. Zahariev, Computing behavior of finite fuzzy machines – algorithm and its application to reduction and minimization, Inf. Sci. 178 (2008) 4152–4165.
[51] T. Petković, Congruences and homomorphisms of fuzzy automata, Fuzzy Sets Syst. 157 (2006) 444–458.
[52] D.W. Qiu, Automata theory based on completed residuated lattice-valued logic (I), Sci. China Ser. E 44 (6) (2001) 419–429.
[53] D.W. Qiu, Pumping lemma in automata theory based on complete residuated lattice-valued logic: a note, Fuzzy Sets Syst. 157 (2006) 2128–2138.
[54] D.W. Qiu, F.C. Liu, Fuzzy discrete-event systems under fuzzy observability and a test algorithm, IEEE Trans. Fuzzy Syst. 17 (3) (2009) 578–589.
[55] L.M. Reyneri, An introduction to fuzzy state automata, in: J. Mira, et al. (Eds.), Biological and Artificial Computation: From Neuroscience to Technology, IWANN 1997, in: Lecture Notes in Computer Science, vol. 1240, 1997, pp. 273–283.
[56] G.G. Rigatos, Fault detection and isolation based on fuzzy automata, Inf. Sci. 179 (2009) 1893–1902.
[57] R. Sabbadin, A possibilistic model for qualitative sequential decision problems under uncertainty in partially observable environments, in: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, UAI 1999, 1999, pp. 567–574.
[58] V. Snášel, A. Keprt, A. Abraham, A.E. Hassanien, Approximate string matching by fuzzy automata, in: K.A. Cyran, et al. (Eds.), Man–Machine Interactions, in: AISC, vol. 59, 2009, pp. 281–290.
[59] A. Stamenković, M. Ćirić, J. Ignjatović, Reduction of fuzzy automata by means of fuzzy quasi-orders, Inf. Sci. 275 (2014) 168–198.
[60] A. Stamenković, M. Ćirić, J. Ignjatović, Different models of automata with fuzzy states, Facta Univ., Ser. Math. Inform. 30 (3) (2015) 235–253.
[61] F. Steimann, K.-P. Adlassnig, Clinical monitoring with fuzzy automata, Fuzzy Sets Syst. 61 (1994) 37–42.
[62] R. van Glabbeek, B. Ploeger, Five determinization algorithms, in: O.H. Ibarra, B. Ravikumar (Eds.), CIAA 2008, in: Lecture Notes in Computer Science, vol. 5148, 2008, pp. 161–170.
[63] R. van Glabbeek, B. Ploeger, Five determinization algorithms, CS-report 08-14, Eindhoven University of Technology, 2008.
[64] L.H. Wu, D.W. Qiu, Automata theory based on complete residuated lattice-valued logic: reduction and minimization, Fuzzy Sets Syst. 161 (2010) 1635–1656.
[65] Q. Wu, T. Wang, Y. Huang, J. Li, Theory research on a new type fuzzy automaton, in: L. Wang, et al. (Eds.), FSKD 2006, in: Lecture Notes in Computer Science, vol. 4223, 2006, pp. 1–10.
[66] H. Xing, D.W. Qiu, F.C. Liu, Automata theory based on complete residuated lattice-valued logic: pushdown automata, Fuzzy Sets Syst. 160 (2009) 1125–1140.
[67] H. Xing, D.W. Qiu, F.C. Liu, Z.J. Fan, Equivalence in automata theory based on complete residuated lattice-valued logic, Fuzzy Sets Syst. 158 (2007) 1407–1422.
[68] H. Xing, Q. Zhang, K. Huang, Analysis and control of fuzzy discrete event systems using bisimulation equivalence, Theor. Comput. Sci. 456 (2012) 100–111.
[69] S. Yu, Regular languages, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, vol. 1, Springer-Verlag, Berlin, Heidelberg, 1997, pp. 41–110.

Please cite this article in press as: Z. Jančić et al., Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems (2016), http://dx.doi.org/10.1016/j.fss.2015.11.019