Behavior, Equivalence, Reduction and Minimization Software for Finite Intuitionistic Fuzzy Machines

K. Peeva, Zl. Zahariev
Faculty of Applied Mathematics and Informatics
Technical University of Sofia, Bulgaria
Sofia 1000, P. O. Box 384
kgp@tu-sofia.bg, zlatko@tu-sofia.bg

Abstract

We investigate finite intuitionistic fuzzy machines with emphases on computing behavior, establishing equivalence of states and solving reduction and minimization problems. We develop functions for these problems in MATLAB environment. The computational complexity of algorithm is discussed. Testing results and examples are supplied.

Keywords: Finite intuitionistic fuzzy machine, behavior, equivalence of states, reduction, minimization.

1 Introduction

Finite $\max - \min$, $\min - \max$ and $\max -$product fuzzy machines are proposed and first studied in [12] – [16]. An extensive bibliography for them is given in [4], [5], [6], [8].

Santos set equivalence, reduction and minimization problems for finite $\max - \min$ fuzzy machines [14] and for finite $\max -$product fuzzy machines [15]. Theoretical results, computational aspects and software for obtaining behavior matrix, establishing equivalence of states, finding redundant states, are studied in [6], [8].

Finite intuitionistic fuzzy machines (FIFM for short) as introduced in [7] are a natural and reasonable extension of $\max - \min$ and $\min - \max$ fuzzy machines.

In this paper we propose theory providing how to compute behavior of FIFM, how to establish equivalence of its states, how to decide whether an FIFM is in reduced or in minimal form. In Section 2 we give preliminaries from intuitionistic fuzzy relational calculus. Computing behavior of FIFM is subject of Sections 3, 4. In Section 5 we investigate equivalence, reduction and minimization. We give algorithm how to extract and compute a representative finite behavior matrix from the complete behavior matrix. Section 6 describes software for computing behavior matrix of FIFM and its implementation for equivalence, reduction and minimization. In Section 7 code snippets show some essential parts of the source code.

2 Intuitionistic Fuzzy Relational Calculus–Basic Notions

The terminology for intuitionistic fuzzy sets is according to [1].

Partial order relation on a partially ordered set (poset) P is denoted by the symbol $\leq$. By a *greatest element* of a poset P we mean an element $b \in P$ such that $x \leq b$ for all $x \in P$. The *least element* of P is defined dually.

Let $\mathbb{I}_* = \langle [\,0,\,1\,],\, \vee,\, \wedge,\ 0,\, 1, \rangle$, where $[\,0,\,1\,]$ is the real closed unit interval, and $\vee$, $\wedge$ are respectively defined by

$$a \vee b = max\{a, b\},\ \ a \wedge b = \min\{a, b\}.$$

$\mathbb{I}_*$ is a complete lattice with universal bounds 0 and 1.

In $\mathbb{I}_*$ the operations α and ε are used. For $a, b \in [0,1]$ they are defined as follows:

$$a\,\alpha\,b = \begin{cases} 1 & \text{if } a \leq b \\ b & a > b \end{cases}, \quad a\,\varepsilon\,b = \begin{cases} b & \text{if } a < b \\ 0 & a \geq b \end{cases}.$$

Operation α is also called Gödel implication and denoted as $\rightarrow_G$ in Gödel algebra and $\langle [\,0,\,1\,],\, \vee,\, \wedge,\ \odot,\, \rightarrow_G,\, 0,\, 1, \rangle$ is residuated lattice [3].

2.1 Matrix products

A matrix $A = \left\langle (\mu_{ij}^A), (\nu_{ij}^A) \right\rangle_{m \times n}$ with $\mu_{ij}^A, \nu_{ij}^A \in [\,0,\,1\,]$ such that $0 \leq \mu_{ij}^A + \nu_{ij}^A \leq 1$ for each $i,\, j, 1 \leq i \leq m,\ 1 \leq j \leq n,\ m, n \in \mathbb{N}$, is called *intuitionistic fuzzy matrix* [7]. In what follows we write 'matrix' or IFM instead of 'intuitionistic fuzzy matrix'.

Two IFM $A_{m\times p}$ and $B_{p\times n}$ are called *conformable*, if the number of columns in A coincides with the number of rows in B.

Several matrix products with conformable matrices may be defined on $\mathbb{I}_*$.

Definition 1.

i) The matrix $C = A * B = \left\langle (\mu_{ij}^C), (\nu_{ij}^C) \right\rangle_{m\times n}$ is called *intuitionistic product* of $A = \left\langle (\mu_{ij}^A), (\nu_{ij}^A) \right\rangle_{m\times n}$ and $B = \left\langle (\mu_{ij}^B), (\nu_{ij}^B) \right\rangle_{m\times n}$ if

$$\mu_{ij}^C = \underset{k=1}{\overset{p}{\vee}}(\mu_{ik}^A \wedge \mu_{kj}^B), \;\; \nu_{ij}^C = \underset{k=1}{\overset{p}{\wedge}}(\nu_{ik}^A \vee \nu_{kj}^B) \;\text{ when }\; 1 \le i \le m, \; 1 \le j \le n.$$

ii) The matrix $C = A \oslash B = \left\langle (\mu_{ij}^C), (\nu_{ij}^C) \right\rangle_{m\times n}$ is called $\alpha - \varepsilon$ product of $A = \left\langle (\mu_{ij}^A), (\nu_{ij}^A) \right\rangle_{m\times n}$ and $B = \left\langle (\mu_{ij}^B), (\nu_{ij}^B) \right\rangle_{m\times n}$ if

$$\mu_{ij}^C = \underset{k=1}{\overset{p}{\wedge}}(\mu_{ik}^A \alpha \mu_{kj}^B), \;\; \nu_{ij}^C = \underset{k=1}{\overset{p}{\vee}}(\nu_{ik}^A \epsilon \nu_{kj}^B) \;\text{ when }\; 1 \le i \le m, \; 1 \le j \le n.$$

The $\alpha - \varepsilon$ product of $A = \left\langle (\mu_{ij}^A), (\nu_{ij}^A) \right\rangle_{m\times n}$ and $B = \left\langle (\mu_{ij}^B), (\nu_{ij}^B) \right\rangle_{m\times n}$ may not be intuitionistic fuzzy matrix.

2.2 Direct and inverse problems

If the IFM $A_{m\times p}$ and $B_{p\times n}$ are given, computing their product is called ***direct problem resolution***. Codes for direct problem resolution, available free under General Public License (GPL), are given in [8].

If $A_{m\times p}$ and $C_{m\times n}$ are given, computing the unknown matrix $B_{p\times n}$ such that $A * B = C$ is is called ***inverse problem resolution***.

Theorem 1 Let $A_{m\times p}$ and $C_{m\times n}$ be given IFM and let $\mathbb{B}_*$ be the set of all IFM such that $A * B = C$. Then $\mathbb{B}_* \neq \emptyset$ iff $A^t \oslash C \in \mathbb{B}_*$.

We develop suitable software based on Theorem 1 to compute behavior of FIFM.

3 Finite intuitionistic fuzzy machines

In this section we define finite intuitionistic fuzzy machine $\mathcal{A}$ over $\mathbb{I}_*$ and describe its complete behavior by suitable matrix $T_{\mathcal{A}}$. We give examples for computing $T_{\mathcal{A}}$ for words in fixed length. Computations are made in two different ways: following the theoretical background and with the functions developed by the authors.

For a finite set C we denote by $|C|$ its cardinality.

Definition 2. [7] A *finite intuitionistic fuzzy machine* (FIFM) over $\mathbb{I}_*$ is a quadruple

$$\mathcal{A} = (X,\, Q,\, Y,\, \mathcal{M}),$$

where:

i) X, Q, Y are nonempty finite sets of input letters, states and output letters, respectively.

ii) $\mathcal{M}$ is the set of transition-output intuitionistic fuzzy matrices of $\mathcal{A}$, that determines its stepwise behavior. Each matrix $M(x|y) = \langle \mu_{qq'}(x|y), \nu_{qq'}(x|y) \rangle \in \mathcal{M}$ is a square matrix of order $|Q|$ and $x \in X,\ y \in Y,\ q, q' \in Q,\ \mu_{qq'}(x|y), \nu_{qq'}(x|y) \in [0,1]$ with $0 \leq \mu_{qq'}(x|y) + \nu_{qq'}(x|y) \leq 1$.

We regard $\mu_{qq'}(x|y)$ as the degree of membership and $\nu_{qq'}(x|y)$ as the degree of non-membership that the FFM will enter state $q' \in Q$ and produce output $y \in Y$ given that the present state is $q \in Q$ and the input is $x \in X$.

3.1 Extended input-output behavior of FIFM

In this subsection we will be interested in operating of $\mathcal{A}$ for words, i. e. for more than one consecutive steps.

The *free monoid of the words* over the set X is denoted by X^* with the *empty word* e as the identity element. If $X \neq \emptyset$ then X^* is countably infinite. The *length of the word* u is denoted by $|\,u\,|$. By definition $|\,e\,| = 0$. Obviously $|\,u\,| \in \mathbb{N}$ for each $u \neq e$.

For $u \in X^*$ and $v \in Y^*$, if $|\,u\,| = |\,v\,|$, we write $(u|v) \in (X|Y)^*$, to distinguish it from the case $(u, v) \in X^* \times Y^*$. We denote by $(X\,|Y)^*$ the set of all input-output pairs of words of the same length:

$$(X\,|Y)^* = \{\, (u|v) \mid u \in X^*,\ v \in Y^*,\ |\,u\,| = |\,v\,| \,\}.$$

Definition 3. Let $\mathcal{A} = (X, Q, Y, \mathcal{M})$ be FIFM over $\mathbb{I}_*$. For any $(u|v) \in (X|Y)^*$ the *extended input-output behavior* of $\mathcal{A}$ upon the law of composition $*$ is determined by the square matrix $M(u|v)$ of order $|\,Q\,|$:

$$M(u\mid v) = \begin{cases} M(x_1\,|y_1) * \ldots * M(x_k\,|y_k), \\ \qquad \text{if} \quad (u|v) = (x_1 \ldots x_k | y_1 \ldots y_k),\ k \geq 1 \\ \\ U, \qquad \text{if} \quad (u|v) = (e|e) \end{cases} \tag{1}$$

where $U = (\left\langle \mu_{ij}^U, \nu_{ij}^U \right\rangle)$ is the square matrix of order $|Q|$ with elements:

$$\langle \mu_{ij}^U, \nu_{ij}^U \rangle = \begin{cases} \langle 1, 0 \rangle & \text{if } i = j \\ \langle 0, 1 \rangle & \text{if } i \neq j \end{cases} .$$

We regard each $\langle \mu_{qq'}(u|v), \nu_{qq'}(u|v) \rangle$ in $M(u|v)$ as degree of membership, degree of non-membership pair that the FIFM $\mathcal{A}$ will enter state $q' \in Q$ and produce output word $v \in Y^*$ under the input word $u \in X^*$ beginning at state $q \in Q$, after $|u| = |v|$ consecutive steps.

3.2 Complete input-output behavior

The complete input-output behavior of $\mathcal{A}$ is determined by column-matrices $T(u|v)_{|Q| \times 1}$ as follows.

$$T(u|v)_{|\,Q\,| \times 1} = (t_q(u|v)) = \begin{cases} M(u|v) * E, \text{ if } (u|v) \neq (e|e); \\ E, \text{ if } (u|v) = (e|e) \end{cases} , \qquad (2)$$

where E is the $|\,Q\,| \times 1$ column-matrix with all elements equal to $\langle 1, 0 \rangle$.

Each element $t_q(u|v)$ of $T(u|v)$ in (2) determines the operation of $\mathcal{A}$ under the input word u beginning at state q and producing the output word v after $|u| = |v|$ consecutive steps.

The *complete behavior matrix* $T_{\mathcal{A}}$ of $\mathcal{A}$ is semi-infinite matrix with $|Q|$ rows and with columns $T(u|v)$, $(u|v) \in (X|Y)^*$, computed with (2) and ordered according to the lexicographical order in $(X|Y)^*$, see Table 1 (if $X \neq \emptyset$ and $Y \neq \emptyset$ then X^* and Y^* are countably infinite and $(X|Y)^*$ is lexicographically ordered).

Table 1: $T_{\mathcal{A}}$ – initial fragment

	$T(e\|e)$	$T(x_1\|y_1)\,...$	...	$T(u\|v)\,...$	...
q_1	$\langle 1, 0 \rangle$	$t_{q_1}(x_1\|y_1)\,...$	...	$t_{q_1}(u\|v)\,...$	...
q_2	$\langle 1, 0 \rangle$	$t_{q_2}(x_1\|y_1)\,...$	...	$t_{q_2}(u\|v)\,...$	...
...	...	...	...	...	...
q_n	$\langle 1, 0 \rangle$	$t_{q_n}(x_1\|y_1)\,...$	...	$t_{q_n}(u\|v)\,...$	...
length l	$l = 0$	$l = 1$	...	$l = \|\,u\,\|$	...

Example 1 Compute initial fragment from $T_{\mathcal{A}}$ if the FIFM $\mathcal{A} = (X, Q, Y, \mathcal{M})$ is given with the following data:

$$X = \{\, x_1, x_2 \,\}, \quad Y = \{\, y_1, y_2 \,\}, \quad Q = \{\, q_1, q_2, q_3 \,\},$$

$$m1 = M(x_1|y_1) = \begin{pmatrix} \langle 0.6, 0.4\rangle & \langle 0.2, 0.8\rangle & \langle 0.5, 0.5\rangle \\ \langle 0.1, 0.9\rangle & \langle 0.6, 0.4\rangle & \langle 0.5, 0.5\rangle \\ \langle 0.2, 0.8\rangle & \langle 0, 1\rangle & \langle 0.1, 0.9\rangle \end{pmatrix}, \quad (3)$$

$$m2 = M(x_1|y_2) = \begin{pmatrix} \langle 0, 0.9\rangle & \langle 0, 0.8\rangle & \langle 0, 1\rangle \\ \langle 0, 0.8\rangle & \langle 0, 0.9\rangle & \langle 0, 1\rangle \\ \langle 0.15, 0.8\rangle & \langle 0.2, 0.8\rangle & \langle 0.1, 0.8\rangle \end{pmatrix}, \quad (4)$$

$$m3 = M(x_2|y_1) = \begin{pmatrix} \langle 0.2, 0.8\rangle & \langle 0.4, 0.5\rangle & \langle 0.1, 0.9\rangle \\ \langle 0.4, 0.5\rangle & \langle 0.3, 0.7\rangle & \langle 0.1, 0.9\rangle \\ \langle 0, 1\rangle & \langle 0, 1\rangle & \langle 0, 1\rangle \end{pmatrix}, \quad (5)$$

$$m4 = M(x_2|y_2) = \begin{pmatrix} \langle 0.3, 0.7\rangle & \langle 0, 0.9\rangle & \langle 0.1, 0.9\rangle \\ \langle 0.3, 0.7\rangle & \langle 0.2, 0.8\rangle & \langle 0.1, 0.9\rangle \\ \langle 0.1, 0.8\rangle & \langle 0.1, 0.8\rangle & \langle 0, 1\rangle \end{pmatrix}. \quad (6)$$

I way. Let

$$T1 = T(x_1|y_1),\ T2 = T(x_1|y_2),\ T3 = T(x_2|y_1),\ T4 = T(x_2|y_2).$$

We implement the above theory and obtain:

$$T1 = \begin{pmatrix} \langle 0.6, 0.4\rangle \\ \langle 0.6, 0.4\rangle \\ \langle 0.2, 0.8\rangle \end{pmatrix}, \quad T2 = \begin{pmatrix} \langle 0, 0.8\rangle \\ \langle 0, 0.8\rangle \\ \langle 0.2, 0.8\rangle \end{pmatrix},$$

$$T3 = \begin{pmatrix} \langle 0.4, 0.5\rangle \\ \langle 0.4, 0.5\rangle \\ \langle 0, 1\rangle \end{pmatrix}, \quad T4 = \begin{pmatrix} \langle 0.3, 0.7\rangle \\ \langle 0.3, 0.7\rangle \\ \langle 0.1, 0.8\rangle \end{pmatrix}.$$

Multiplying (3) – (6) by two's and applying (2), we obtain:

$$m11 = M(x_1x_1|y_1y_1) = M(x_1|y_1) * M(x_1|y_1) = \begin{pmatrix} \langle 0.6, 0.4\rangle & \langle 0.2, 0.5\rangle & \langle 0.5, 0.5\rangle \\ \langle 0.2, 0.8\rangle & \langle 0.6, 0.4\rangle & \langle 0.5, 0.5\rangle \\ \langle 0.2, 0.8\rangle & \langle 0.2, 0.8\rangle & \langle 0.2, 0.8\rangle \end{pmatrix},$$

Table 2: An initial fragment from $T_{\mathcal{A}}$ for Example 2

	$T0$	$T1$	$T2$	$T3$	$T4$	$T11$	...
q_1	$\langle 1,0\rangle$	$\langle 0.6,0.4\rangle$	$\langle 0,0.8\rangle$	$\langle 0.4,0.5\rangle$	$\langle 0.3,0.7\rangle$	$\langle 0.6,0.4\rangle$	...
q_2	$\langle 1,0\rangle$	$\langle 0.6,0.4\rangle$	$\langle 0,0.8\rangle$	$\langle 0.4,0.5\rangle$	$\langle 0.3,0.7\rangle$	$\langle 0.6,0.4\rangle$	...
q_3	$\langle 1,0\rangle$	$\langle 0.2,0.8\rangle$	$\langle 0.2,0.8\rangle$	$\langle 0,1\rangle$	$\langle 0.1,0.8\rangle$	$\langle 0.2,0.8\rangle$	...

$$T11 = T(x_1x_1|y_1y_1) = \begin{pmatrix} \langle 0.6,0.4\rangle \\ \langle 0.6,0.4\rangle \\ \langle 0.2,0.8\rangle \end{pmatrix}, \text{ etc.}$$

It is obvious that this method of computation is tremendous even for words in small length. This motivated us to develop functions for computing $T_{\mathcal{A}}$, as explained in II way.

II way. We develop function in MATLAB workspace and software for computing the initial fragment T_k of the matrix $T_{\mathcal{A}}$ for words in fixed length $\leq k$ for finite intuitionistic fuzzy machines. The function (described in Section 6) is:

find_t_int(m, word_length) where

- **m** is a cell array with the initial data intuitionistic matrices from the set $\mathcal{M}$ of the FIFM.
- **word_length** is the desired word length k. It also gives the number of executive steps for finding T_k.

For Example 1 the function **find_t_int(m,2)** results (compare next columns 1 – 6 with these in Table 2):

```
>> find_t_int(m,2)
step=1; time=0.085189; t=

b =
Columns 1 through 4
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.20, 0.80>   <0.20, 0.80>   <0.00, 1.00>

Column 5
<0.30, 0.70>
```

```
<0.30, 0.70>
<0.10, 0.80>

step=2; time=0.074356; t=
b =
Columns 1 through 4
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.20, 0.80>   <0.20, 0.80>   <0.00, 1.00>

Columns 5 through 8
<0.30, 0.70>   <0.60, 0.40>   <0.20, 0.80>   <0.40, 0.50>
<0.30, 0.70>   <0.60, 0.40>   <0.20, 0.80>   <0.40, 0.50>
<0.10, 0.80>   <0.20, 0.80>   <0.10, 0.80>   <0.20, 0.80>

Columns 9 through 12
<0.30, 0.70>   <0.00, 0.80>   <0.00, 0.80>   <0.00, 0.80>
<0.30, 0.70>   <0.00, 0.80>   <0.00, 0.80>   <0.00, 0.80>
<0.20, 0.80>   <0.20, 0.80>   <0.10, 0.80>   <0.20, 0.80>

Columns 13 through 16
<0.00, 0.80>   <0.40, 0.50>   <0.10, 0.80>   <0.40, 0.50>
<0.00, 0.80>   <0.40, 0.50>   <0.10, 0.80>   <0.40, 0.50>
<0.20, 0.80>   <0.00, 1.00>   <0.00, 1.00>   <0.00, 1.00>

Columns 17 through 20
<0.30, 0.70>   <0.30, 0.70>   <0.10, 0.80>   <0.30, 0.70>
<0.30, 0.70>   <0.30, 0.70>   <0.10, 0.80>   <0.30, 0.70>
<0.00, 1.00>   <0.10, 0.80>   <0.00, 0.80>   <0.10, 0.80>

Column 21
<0.30, 0.70>
<0.30, 0.70>
<0.10, 0.80>
```

The function **find_t_(m, word_length)** is based on the notions and theoretical results for finite intuitionistic fuzzy machines. It automatizes all operations described in the I way of Example 1.

4 Behavior Matrix

The complete input-output behavior matrix $T_{\mathcal{A}}$ of any FIFM $\mathcal{A}$ is semi-infinite – it has finite number of rows (equal to the number of states in Q) and infinite number of columns. Many conventional problems require to extract from $T_{\mathcal{A}}$ a finite submatrix $B_{\mathcal{A}}$ (called behavior matrix) that provides solving these problems.

In this section we propose algorithm for extracting a finite behavior matrix $B_{\mathcal{A}}$ from the complete behavior matrix $T_{\mathcal{A}}$. $B_{\mathcal{A}}$ is expected to have sufficiently good properties for solving equivalence, reduction and minimization problems.

In order to explain computing $B_{\mathcal{A}}$ we provide supplementary information – what is intuitionistic linear combination, how to establish that a vector is intuitionistic linear combination of a set of vectors and how do we implement this for obtaining $B_{\mathcal{A}}$.

4.1 Intuitionistic Linear Combination

Let $A(1) = \left(\left\langle \mu^{A(1)}, \nu^{A(1)} \right\rangle\right)_{n\times 1}, \ldots, A(k) = \left(\left\langle \mu^{A(k)}, \nu^{A(k)} \right\rangle\right)_{n\times 1}$ be intuitionistic fuzzy column-vectors.

Definition 4. We say that a column-vector $C_{n\times 1}$ is ***intuitionistic linear combination*** of the vectors $A(i)_{n\times 1}$, $1 \leq i \leq k$, with coefficients $\langle x_i, y_i \rangle \in [0,\, 1] \times [0,\, 1]$, with $0 \leq x_i + y_i \leq 1$, $1 \leq i \leq k$, if

$$C = \left\langle \left(\mu^{A(1)} \wedge x_1\right) \vee \cdots \vee \left(\mu^{A(k)} \wedge x_k\right), \left(\nu^{A(1)} \vee y_1\right) \wedge \cdots \wedge \left(\mu^{A(k)} \vee y_k\right) \right\rangle.$$

Checking whether $C_{n\times 1}$ is intuitionistic linear combination of $A(1)$, $\ldots, A(k)$ requires to solve the system

$$A * X = C$$

for the unknown X, if A and C are given. When the system is consistent (inconsistent, respectively) then the right-hand side vector is (is not, respectively) intuitionistic linear combination of the vectors forming the matrix of coefficients.

The finite behavior matrix $B_{\mathcal{A}}$ contains only linearly independent columns from $T_{\mathcal{A}}$. Hence, when computing $B_{\mathcal{A}}$ from $T_{\mathcal{A}}$, each column from $T_{\mathcal{A}}$ that is a linear combination of the previous columns should be removed. For checking whether a column is intuitionistic linear combination we develop software that implements functions from [8] for solving fuzzy linear system of equations (inverse problem resolution).

4.2 Algorithm for computing behavior matrix $B_{\mathcal{A}}$

We denote by $T(i)$ the finite submatrix of $T_{\mathcal{A}}$ containing the columns $T(u|v)$ for the words of length not greater than i, $i \in \mathbb{N}$. Let $B(i)$ be a submatrix of $T(i)$ obtained by omitting all columns from $T(i)$ that are linear combination of the previous columns.

For arbitrary matrices C and D we write $C \subseteq D$, if each column of C is a column of D. If each column in D is a linear combination of the columns from C, we write $C \cong D$. Obviously for each $i \in \mathbb{N}$, we have [6]:

1. $T(i) \subseteq T(i+1) \subseteq \ldots \subseteq T$;
2. $B(i) \subseteq B(i+1) \subseteq \ldots \subseteq B$;
3. $B(i) \subseteq T(i)$.

For any FIFM $\mathcal{A}$ we can obtain the matrix $B(i)$ from $T(i)$ – it suffers to remove the columns from $T(i)$ that are linear combination of the previous columns. This is possible because we have method and software to solve $\max - \min$ and $\min - \max$ fuzzy linear system of equations [8] we develop functions for establishing $\max - \min$ and $\min - \max$ linear dependence or linear independence [9], [10] and finally we develop functions for computing $B(i)$, see next section.

Definition 5. The matrix $B_{\mathcal{A}}$ obtained by omitting all columns from $T_{\mathcal{A}}$ that are intuitionistic linear combination of the previous columns is called **behavior matrix** for $\mathcal{A}$.

Theorem 3. For any FIFM $\mathcal{A}$ the following statements hold:

1. There exists an integer k, such that $T(k) = T(k+1)$ and $B(k) = B_{\mathcal{A}}$.
2. If $T(k) \cong T(k+1)$, then:
   - $T(k) \cong T(k+p) \cong \ldots T_{\mathcal{A}}$ for each $p = 1, \ldots$;
   - $B(k) = B(k+p) = \ldots B_{\mathcal{A}}$ for each $p = 1, \ldots$;
3. $B_{\mathcal{A}} \cong T_{\mathcal{A}}$.

The proof is based on the validity of the results for $\max - \min$ FFM [6], their validity for $\min - \max$ FFM follows by dualization principle – $(\vee, \wedge, \neg)$ is a dual triple [2] and product of two intuitionistic fuzzy matrices is intuitionistic fuzzy matrix also.

Algorithm for computing the behavior matrix $B_{\mathcal{A}}$ for FIFM $\mathcal{A} = (X, Q, Y, \mathcal{M})$.

1. Enter the set of matrices $\mathcal{M}$.

2. Find k, such that $T(k) \cong T(k+1)$.

3. Obtain $B(k) = B_{\mathcal{A}}$ excluding all linear combinations from $T(k)$.

4. End.

It is established in [6] that the behavior matrix $B_{\mathcal{A}}$ is finite for $\max - \min$ finite fuzzy machine $\mathcal{A} = (X, Q, Y, \mathcal{M})$ and the time complexity function for computing $B_{\mathcal{A}}$ is exponential. The same is valid for $\min - \max$ finite fuzzy machine. This provides that the behavior matrix $B_{\mathcal{A}}$ is finite for FIFM. The algorithm for finding behavior matrix $B_{\mathcal{A}}$ has exponential time complexity.

Example 1 – continued. Compute the behavior matrix for the max-min FIFM given in Example 2. The function **find_b_int(m)** results that $T(1) = T(2)$ and hence $B = T(1)$:

```
>> find_b_int(m)
step=1; time=0.69697; t=

b =
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.20, 0.80>   <0.20, 0.80>   <0.00, 1.00>

step=2; time=0.014771; t=

b =
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.20, 0.80>   <0.20, 0.80>   <0.00, 1.00>

ans =
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.20, 0.80>   <0.20, 0.80>   <0.00, 1.00>
```

5 Equivalence of States, Reduction, Minimization

We demonstrate in this section how the software for finding $T_{\mathcal{A}}$ and $B_{\mathcal{A}}$ can be implemented for solving equivalence problems.

Definition 6. The states $q_i \in Q$ and $q_j \in Q$ in FIFM $\mathcal{A} = (X, Q, Y, \mathcal{M})$ are called **equivalent** if the input-output behavior of $\mathcal{A}$ when begin with state q_i is the same as its input-output behavior when begin with state q_j.

It means that the $i-$th and $j-$th rows are identical in $T_{\mathcal{A}}$ and in $B_{\mathcal{A}}$. Since $T_{\mathcal{A}}$ is semi-infinite, we can not derive from it equivalence of states. In order to solve the problem we have to extract $B_{\mathcal{A}}$ from $T_{\mathcal{A}}$ and to check $B_{\mathcal{A}}$ for identical rows.

Definition 7. FIFM $\mathcal{A} = (X, Q, Y, \mathcal{M})$ is in **reduced form** if there does not exist equivalent states.

Function **find_b_int(m, cond)** establishes equivalence of states and whether the FIFM is in reduced form, if cond = 'reduce'.

Example 1 continued. The machine with data in Example 1 is not in reduced form:

```
>> find_b_int(m, 'reduce')
step=1; time=0.0040656; t=

b =
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.20, 0.80>   <0.20, 0.80>   <0.00, 1.00>
step=2; time=0.014863; t=

b =
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.20, 0.80>   <0.20, 0.80>   <0.00, 1.00>
```

Since the first and the second row in $B_{\mathcal{A}}$ are identical, the states q_1 and q_2 are equivalent and $\mathcal{A} = (X, Q, Y, \mathcal{M})$ is not in reduced form. The behavior matrix of the reduced form FIFM follows:

```
ans =
<1.00, 0.00>   <0.60, 0.40>   <0.00, 0.80>   <0.40, 0.50>
<1.00, 0.00>   <0.20, 0.80>   <0.20, 0.80>   <0.00, 1.00>
```

An FIFM $\mathcal{A} = (X, Q, Y, \mathcal{M})$ is in **minimal form** if the input-output behavior of $\mathcal{A}$ when begin with initial state q_i is the same as its input-output behavior when begin with so called isolating initial distribution of membership degrees over Q – it means that the $i-$th row of $T_{\mathcal{A}}$ is a linear combination of the other rows. For any FIFM $\mathcal{A} = (X, Q, Y, \mathcal{M})$ with known behavior matrix $B_{\mathcal{A}}$, it is algorithmically solvable whether $\mathcal{A}$ is in minimal form (for $\max - \min$ finite fuzzy machines see [6], take into account dualization principle and stability of intuitionistic multiplication).

Function **find_b_int(m, cond)** establishes whether the FIFM is in minimal form, if cond = 'minimize'.

6 Software Description

As mentioned, the main functionality of this software is concentrated in **find_t_int()** and **find_b_int()** functions. The substantial difference between them is that **find_t_int()** computes complete behavior matrix for a given fuzzy machine and therefore it does not remove the columns that are linear combination of the other columns, while **find_b_int()** removes these columns. Function **find_b_int()** can also reduce or minimize the computed behavior matrix, while **find_t_int()** does not do this due a theoretical reasons. Their parameters are described below.

For **find_t_int()** these are:

- **m** - a cell array with all matrices from $\mathcal{M}$ representing FIFM behavior for words in length 1. All matrices are instances from class 'im'. It have not default value, but if this parameter is omitted a dialogue with the user will be done to receive the matrices.
- **word_length** - the maximum word length for which we compute the complete behavior matrix. This parameter is mandatory.

The function **find_t_int()** returns the complete behavior matrix T_k, where k is the desired word length.

The parameters for **find_b_int** are:

- **m** - as above
- **cond** - this parameter can be one of 'none', 'minimize' or 'reduce'. It defines do we want to make minimization ('minimize') or reduction ('reduce') to the final behavior matrix or leave it as it is ('none'). Default value is 'none'.

- **word_length** - in this function this parameter is optional. The default value is '-1' which is used for 'unlimited'. In this case the function returns the behavior matrix for the given fuzzy machine for word with any length.

This algorithm has exponential time complexity and exponential memory complexity. If we have $|X|$ input letters and $|Y|$ output letters, we produce $(|X|.|Y|)^k$ matrices, where k is the length of the word.

7 Code Snippets

All configuration data are organized in separated arrays:

```
compositions = {'maxmin'};
compositions_int = {'minmax'};
norms = {'max', 'min'};
conds = {'none', 'reduce', 'minimize'};
```

Presented below function obtains all matrices for words in length $k + 1$. All matrices are saved in the same cell array which we first expand to the new size and then fill backwards. We make this to save memory (in this manner we do not keep in to memory matrices for the k^{th} step and matrices for the $(k + 1)^{th}$ step, but just the matrices for the $(k + 1)^{th}$ step).

```
function b=next_stage(m,b)
global composition_key;
global compositions;
global compositions_int;

length_m=numel(m);
length_b=size(b,2);
k=length_m*length_b;
for i=length_m:-1:1
    for j=length_b:-1:1
        function_name1 = ['fuzzy_'...
                ... compositions{composition_key}];
        function_name2 = ['fuzzy_'...
                ... compositions_int{composition_key}]';
        bj = b(:,j);
        mi = m{i};
```

```
        bm = feval(function_name1, mi.m, bj.m);
        bn = feval(function_name2, mi.n, bj.n);
        b(:,k) = im(bm,bn);
        k=k-1;
    end;
end;
```

Next function is actually building the behavior matrix in function **find_b_int()**. It takes all vectors $T(i|j)$ for the current step, checks any single one of them if it is linear combination of the previous vectors and vector is added to the behavior matrix if it is not linear combination of the previous vectors.

```
function b=find_bn(cols,b)
global composition_key;
global compositions;
function_name =
    ['is_fuzzy_' compositions{composition_key} '_lc_int'];
for i=1:size(cols,2)
    bc=cols(:,i);
    is_comb = feval(function_name, b, bc);
    if ~(isa(is_comb, 'im') || is_comb == true)
        b(:, size(b,2)+1) = bc;
    end;
end;
```

8 Conclusions

All other equivalence, reduction and minimization problems as introduced in [6] are solved by suitable functions, developed by the authors. We give here illustration only for some of them. The reason is that the theoretical background for these problems is tremendous, see [5], [6] and this will embarrass the reader. The essence is that we propose software (under the request to the authors) for finding behavior matrix and solving equivalence, reduction and minimization problems for FIFM.

References

[1] K. Atanassov (1999), Intuitionistic Fuzzy Sets, Theory and Applications, Physica-Verlag.

[2] B. De Baets (2000), Analytical solution methods for fuzzy relational equations, in D. Dubois and H. Prade (eds.), the series: *Fundamentals of Fuzzy Sets, The Handbooks of Fuzzy Sets Series*, Vol. 1, Kluwer Academic Publishers, 291-340.

[3] A. Di Nola, A. Lettieri, I. Perfilieva, V. Novák (2007), Algebraic analysis of fuzzy systems, *Fuzzy Sets and Systems*, **158** (1) 1 - 22.

[4] G. Klir, B. Yuan (1995), Fuzzy Sets and Fuzzy Logic: Theory and Applications, Prentice Hall PTR, NJ.

[5] J. N. Mordeson and D. S. Malik (2002), *Fuzzy Automata and Languages – Theory and Applications*, CAPMAN&HALL/CRC, London.

[6] K. Peeva (2004), Finite L-Fuzzy Machines *Fuzzy Sets and Systems*, Vol 141, No 3, pp. 415-437.

[7] K. Peeva (2003), Finite Intuitionistic Fuzzy Machines, *Notes on IFS*, volume 9 Number 3 pp 40- 45 *Proceedings of the Seventh International Conference on Intuitionistic Fuzzy Sets, Sofia, 23-24 August 2003.*

[8] K. Peeva and Y. Kyosev (2004), *Fuzzy Relational Calculus-Theory, Applications and Software (with CD-ROM)*, In the series *Advances in Fuzzy Systems - Applications and Theory*, Vol. 22, World Scientific Publishing Company. CD-ROM http://mathworks.net

[9] K. Peeva, Zl. Zahariev (2006), Linear dependence in fuzzy algebra, in *Proceedings of 31th International Conference AME, Sozopol June 2005 (in press).*

[10] K. Peeva, Zl. Zahariev (2006), Software for Testing Linear Dependence in Fuzzy Algebra, in *Proceedings of Second International Scientific Conference Computer Science, Chalkidiki, 30 Sept -2 Oct 2005*, part I, pp 294-299.

[11] E. Sanchez (1976), Resolution of composite fuzzy relation equations, *Information and Control*, 30 38-48.

[12] E. S. Santos (1968), Maximin automata, *Information and Control*, 13 363-377.

[13] E. S. Santos (1968), Maximin, minimax and composite sequential mashines, *J. Math. Anal. Appl.*, 24 246-259.

[14] E. S. Santos (1972), On reduction of maxi-min machines, *J. Math. Anal. Appl.*, 40 60-78.

[15] E. S. Santos (1972), Max-product machines, *Journal of Math. Anal. And Applications*, 37 677-686.

[16] E. S. Santos and W. G. Wee (1968), General formulation of sequential machines, *Information and Control*, 12(1) 5-10.