Bisimulations for weighted automata over an additively idempotent semiring

Nada Damljanović, Miroslav Ćirić, Jelena Ignjatović

University of Kragujevac, Faculty of Technical Sciences in Čačak, Svetog Save 65, P. O. Box 131, 32000 Čačak, Serbia
University of Niš, Faculty of Sciences and Mathematics, Višegradska 33, P. O. Box 224, 18000 Niš, Serbia

Abstract

We show that the methodology for testing the existence and computation the greatest simulations and bisimulations, developed in the framework of fuzzy automata [M. Ćirić et al., Fuzzy Sets and Systems 186 (2012) 100–139, 208 (2012) 22–42], can be applied in a similar form to weighted automata over an additively idempotent semiring. We define two types of simulations and four types of bisimulations for weighted automata over an additively idempotent semiring as solutions to particular systems of matrix inequalities, and we provide polynomial-time algorithms for deciding whether there is a simulation or bisimulation of a given type between two weighted automata, and for computing the greatest one, if it exists. The algorithms are based on the concept of relative Boolean residuation for matrices over an additively idempotent semiring, that we introduce here. We also prove that two weighted automata $A$ and $B$ are forward bisimulation equivalent, i.e., there is a row and column complete forward bisimulation between them, if and only if the factor weighted automata with respect to the greatest forward bisimulation equivalence matrices on $A$ and $B$ are isomorphic. In addition, we show that the factor weighted automaton with respect to the greatest forward bisimulation equivalence matrix on a weighted automaton $A$ is the unique (up to an isomorphism) minimal automaton in the class of all weighted automata which are forward bisimulation equivalent to $A$.

Keywords: Weighted automaton; Additively idempotent semiring; Boolean matrix; Simulation; Bisimulation; Factor weighted automaton.

1. Introduction

Simulation and bisimulation relations are a very powerful tool that has been used in many areas of computer science to match moves and compare the behavior of various systems, as well as to reduce the number of states of these systems. They have been introduced by Milner [40] and Park [42] in computer science, i.e., in concurrency theory, but roughly at the same time they have been also discovered in some areas of mathematics, e.g., in modal logic and set theory. Afterwards, the use of simulation and bisimulation has gained a long and rich history and their various forms have been defined and applied to different systems. They are employed today in the study of functional languages, object-oriented languages, types, data types, domains, databases, compiler optimizations, program analysis, verification tools, etc.

The most common structures on which simulations and bisimulations have been studied are labelled transition systems, but they have also been investigated in the context of deterministic, nondeterministic, weighted, fuzzy, probabilistic, timed, hybrid and other kinds of automata. One type of bisimulations for weighted automata has been introduced by Ésik and Kuich [22] (and for Boolean automata by Bloom and Ésik [7]), under the name simulation. Under the same name this concept has been studied by Ésik and Maletti [23], and under different names in [3–5, 10, 38, 43]. Béal and Perrin [5] used the name backward...
elementary equivalence, Béal, Lombardy, and Sakarovitch [3, 4] the name conjugacy, which originates from applications in symbolic dynamics, and Buchholz [10] used the name (forward) relational simulation. It is important to note that Béal, Lombardy, and Sakarovitch [3, 4] found that a semiring $S$ often has the following property: two weighted automata over $S$ are equivalent if and only if they are connected by a finite chain of simulations. Semirings having this property include the Boolean semiring [7], the semiring of natural numbers and the ring of integers [3, 4], etc. An example of a semiring which does not have this property is the tropical semiring [23]. In addition, the mentioned concept was used in [22] in the completeness proof of the iteration semiring theory axioms for regular languages. Note also that Ésik and Kuich’s simulations, as well as Béal, Lombardy, and Sakarovitch’s conjugacies, were defined as matrices over a semiring which satisfy the same conditions which we use here to define backward-forward bisimulations, one of our four types of bisimulations. Buchholz’s definition uses the same conditions, but in contrast, his bisimulations are defined to be exclusively relational or functional matrices.

A new approach to simulations and bisimulations has been recently proposed in [18, 19], in the framework of fuzzy automata, and in [16] it has been applied to ordinary nondeterministic automata. Two types of simulations and four types of bisimulations for fuzzy automata have been defined as fuzzy relations (fuzzy matrices) which are solutions to certain systems of fuzzy relation (matrix) inequalities. Such an approach can not be directly applied to weighted automata over an arbitrary semiring, because it requires ordering of matrices, and therefore, an ordering in the underlying semiring, and, in general, the underlying semiring does not have to be ordered. On the other hand, even if the underlying semiring is ordered, we have the problem how to check the solvability and find solutions to the corresponding systems of matrix inequalities and compute simulations and bisimulations. Algorithms developed in [19], for testing the existence of simulations and bisimulations between fuzzy automata and computing the greatest ones (when they exist), are strongly based on the concept of residuation for fuzzy matrices, which is an immediate consequence of residuation in the underlying structures of truth values. However, residuation is present only in some special classes of semirings, such as, for example, max-plus and related semirings. The main aim of this paper is to show that these problems can be partially solved when the weights are taken from an additively idempotent semiring. Such semirings possess a natural ordering which allows us to define simulations and bisimulations by means of systems of matrix inequalities. Moreover, the zero and unit of an additively idempotent semiring form a subsemiring isomorphic to the Boolean semiring, and consequently, matrices with entries in this subsemiring can be treated as Boolean matrices, that is, as ordinary binary relations. Finally, although in general there is no residuation for matrices over an additively idempotent semiring, there is some kind of relative residuation that results in a Boolean matrix. This enables us to find solutions of our systems of matrix inequalities in the class of Boolean matrices, and therefore, to develop the theory of simulations and bisimulations based on relational matrices (ordinary binary relations).

Our main results are the following. We define two types of simulations and four types of bisimulations for weighted automata over an additively idempotent semiring, but because of the duality we further consider only one type of simulations and two types of bisimulations. They are defined to be relational matrices. We also introduce the concepts of relative right and left residuals for matrices over an additively idempotent semiring, and prove the corresponding adjunction properties (Theorems 5.1 and 5.2). Using relative residuals we provide an equivalent form of conditions that define forward simulations (Theorem 5.3), which enables to give a polynomial-time algorithm for deciding whether there is a forward simulation between two weighted automata, and whenever there is at least one such simulation, the same algorithm computes the greatest one (cf. Theorem 5.4 and Algorithm 5.5). We also point out how to construct the corresponding algorithms for forward bisimulations and backward-forward bisimulations. Special attention is paid to bisimulations that are uniform Boolean matrices. We show that a uniform Boolean matrix is a forward bisimulation if and only if its kernel and co-kernel are forward bisimulation equivalence matrices and there is a special isomorphism between related factor weighted automata (Theorem 6.1). A similar theorem for backward-forward bisimulations is also given (Theorem 6.8). Furthermore, we prove that if two weighted automata $\mathcal{A}$ and $\mathcal{B}$ are forward bisimulation equivalent, i.e., if there is a uniform forward bisimulation between them, then there is the greatest forward bisimulation between $\mathcal{A}$ and $\mathcal{B}$ and it is a uniform Boolean matrix, whose kernel and co-kernel are respectively the greatest forward bisimulation equivalence matrices.
on $\mathcal{A}$ and $\mathcal{B}$ (cf. Theorem 6.5). Then we get that two weighted automata $\mathcal{A}$ and $\mathcal{B}$ are forward bisimulation equivalent if and only if the factor weighted automata with respect to the greatest forward bisimulation equivalence matrices on $\mathcal{A}$ and $\mathcal{B}$ are isomorphic. Also, we show that the factor weighted automaton of an arbitrary weighted automaton $\mathcal{A}$ with respect to the greatest forward bisimulation equivalence matrix on $\mathcal{A}$ is the unique (up to an isomorphism) minimal automaton in the class of all weighted automata that are forward bisimulation equivalent to $\mathcal{A}$ (Theorem 6.7). Finally, we prove that the concepts of a forward and a backward-forward bisimulation coincide when working with functional matrices (cf. Theorem 6.9).

The structure of the paper is as follows. In Section 2 we introduce basic notions and notation related to additively idempotent semirings and matrices over them, as well as to Boolean matrices. Special attention is paid to uniform Boolean matrices. Then in Section 3 we give basic notions and notation concerning weighted automata. Section 4 contains our definitions of simulations and bisimulations, and results describing their basic properties. In Section 5 we provide algorithms for checking the existence and computing the greatest simulations and bisimulations. At the end, Section 6 contains results that characterize uniform forward and backward-forward bisimulations, and forward bisimulation equivalent weighted automata. It should be noted that the proofs of some theorems presented here are similar to proofs of the corresponding theorems from [16, 18, 19], and such proofs are omitted.

Finally, it is worth to point out that additively idempotent semirings, which are used here for modeling weights, are one of the most important types of semirings with significant applications in many areas of mathematics, computer science, and operation research, e.g., in the theory of automata and formal languages [20, 36], optimization theory [2, 28, 29, 41], idempotent analysis [30, 35], theory of programming languages [6], data analysis, discrete event systems theory [1], algebraic modeling of fuzziness and uncertainty, algebra of formal processes, etc. In particular, applications of additively idempotent semirings include solution to a wide variety of optimal path problems in graphs, extensions of classical algorithms for shortest path problems to a whole class of nonclassical path-finding problems (such as shortest paths with time constraints, shortest paths with time-dependent lengths on the arcs, etc.), solution of various nonlinear partial differential equations, such as Hamilton-Jacobi, and Burgers equations, the importance of which is well-known in physics, etc.

2. Preliminaries

Throughout this paper, $\mathbb{N}$ denotes the set of natural numbers (without zero), $X^+$ and $X^*$ denote respectively the free semigroup and the free monoid over an alphabet $X$, and $\epsilon$ denotes the unit in $X^*$. Also, for an arbitrary assertion $\Psi$ in the classical Boolean logic let $[\Psi]$ denote its truth value, i.e., $[\Psi] = 1$ if $\Psi$ is true, and $[\Psi] = 0$ if $\Psi$ is false.

2.1. Semirings

A semiring is a structure $(S, +, \cdot, 0, 1)$ consisting of a set $S$, two binary operations $+$ and $\cdot$ on $S$, and two constants $0, 1 \in S$ such that the following is true:

(i) $(S, +, 0)$ is a commutative monoid,
(ii) $(S, \cdot, 1)$ is a monoid,
(iii) the distributivity laws $(a + b) \cdot c = a \cdot c + b \cdot c$ and $c \cdot (a + b) = c \cdot a + c \cdot b$ hold for every $a, b, c \in S$,
(iv) $0 \cdot a = a \cdot 0 = 0$, for every $a \in S$.

As usual, we identify the structure $(S, +, \cdot, 0, 1)$ with its carrier set $S$. We call $S$ an additively idempotent semiring if, in addition to (i)–(iv), it satisfies the condition

(v) $a + a = a$, for every $a \in S$.

Note that (v) is equivalent to $1 + 1 = 1$ in the presence of (i)–(iv). Additively idempotent semirings are also known as path algebras, because of their important application in the study of the algebraic path problem,
and in many sources they were called just *idempotent semirings*. On an additively idempotent semiring \( S \) we define a partial order by

\[
a \leq b \iff a + b = b, (1)
\]

for arbitrary \( a, b \in S \). Note also that if \( S \) is an additively idempotent semiring, then its additive monoid \((S, +, 0)\) is a join-semilattice (upper semilattice) with zero (cf. \([8, 34]\)).

**Example 2.1.** There are many important examples of additively idempotent semirings. We list the most important ones.

1. The tropical semirings are semirings \((\mathbb{N} \cup \{\infty\}, \min, +, \infty, 0)\) and \((\mathbb{R} \cup \{\infty\}, \min, +, \infty, 0)\), and the arctic semirings are semirings \((\mathbb{N} \cup \{-\infty\}, \max, +, -\infty, 0)\) and \((\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)\). Note that the corresponding tropical and arctic semirings are mutually isomorphic. The semiring \((\mathbb{R} \cup \{\infty\}, \min, +, \infty, 0)\) is also known as the min-plus algebra or the min-plus semiring, and \((\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)\) is known as the max-plus algebra or the max-plus semiring.

2. The Viterbi semiring (also called the probabilistic semiring) is the semiring \([(0, 1], \max, \cdot, 0, 1)\).

3. A two-element additively idempotent semiring is isomorphic to the semiring \([(0, 1], \max, \min, 0, 1)\). It is called the Boolean semiring.

4. The semiring \((2^X, \cup, \cdot, \emptyset, \{\epsilon\})\) of all languages over an alphabet \( X \) (where \( \cdot \) denotes the usual concatenation of languages) is an additively idempotent semiring with language inclusion as its natural ordering.

5. The semiring \((2^{A \times A} \cup \circ, \cdot, \emptyset, \Lambda)\) of all binary relations on a set \( A \) (where \( \circ \) denotes the usual composition of relations and \( \Lambda \) is the equality relation) is an additively idempotent semiring with relation (set) inclusion as its natural ordering.

6. Bounded distributive lattices, semiring-reducts of semilattice ordered monoids and of complete residuated lattices, and Brouwerian lattices are additively idempotent semirings.

For more information on additively idempotent semirings and their applications the readers are referred to \([1, 2, 6, 14, 29, 30, 35]\).

2.2. Matrices

In the rest of this paper, if not noted otherwise, let \( S \) be an additively idempotent semiring.

Let \( A \) and \( B \) be finite non-empty sets. A function \( \mu : A \times B \rightarrow S \) is called an \( A \times B \)-matrix over \( S \), and a function \( v : A \rightarrow S \) is called an \( A \)-vector over \( S \). As is usual when working with functions, we denote the set of all \( A \times B \)-matrices over \( S \) by \( S^{A \times B} \), and the set of all \( A \)-vectors over \( S \) by \( S^A \). We will sometimes identify \( A \)-vectors over \( S \) with \( C \times A \)-matrices or \( A \times C \)-matrices over \( S \), where \( |C| = 1 \). For a matrix \( \mu \in S^{A \times B} \), and elements \( a \in A \) and \( b \in B \), by \( \mu(a, \cdot) \) we denote the row-vector corresponding to \( a \), and by \( \mu(\cdot, b) \) the column-vector corresponding to \( b \), i.e., \( \mu(a, \cdot)(y) = \mu(a, y) \) and \( \mu(\cdot, b)(x) = \mu(x, b) \), for all \( y \in B \) and \( x \in A \).

For matrices \( \mu_1, \mu_2 \in S^{A \times B} \), their matrix sum \( \mu_1 + \mu_2 \in S^{A \times B} \) and the ordering \( \mu_1 \leq \mu_2 \) on \( S^{A \times B} \) are defined pointwise. The Hadamard product \( \mu_1 \odot \mu_2 \in S^{A \times B} \) is also defined pointwise, by

\[
(\mu_1 \odot \mu_2)(a, b) = \mu_1(a, b) \cdot \mu_2(a, b), (2)
\]

for every \((a, b) \in A \times B\).

For finite non-empty sets \( A, B \) and \( C \), and matrices \( \mu_1 \in S^{A \times B} \) and \( \mu_2 \in S^{B \times C} \), we define the matrix product \( \mu_1 \cdot \mu_2 \in S^{A \times C} \) by

\[
(\mu_1 \cdot \mu_2)(a, c) = \sum_{b \in B} \mu_1(a, b) \cdot \mu_2(b, c), (3)
\]

for any \((a, c) \in A \times C\). If \( \mu_1 \) is a vector (i.e., \(|A| = 1\)), or \( \mu_2 \) is a vector (i.e., \(|C| = 1\)), then (3) defines matrix-vector products, and if both \( \mu_1 \) and \( \mu_2 \) are vectors (i.e., \(|A| = |C| = 1\)), then (3) defines the scalar
product of $\mu_1$ and $\mu_2$. Since distributivity of the multiplication operation over the addition operation holds, the matrix product and matrix-vector products are associative (whenever they are defined). Note that $(S_{AXB}, +, 0, 1)$ is an additively idempotent semiring, for any additively idempotent semiring $S$ and non-empty set $A$, where $0$ and $1$ respectively denote the zero and the unit matrix. The natural ordering on this additively idempotent semiring, defined by the rule $(1)$, coincides with the above mentioned pointwise ordering of matrices. The transpose of a matrix $\mu$ is denoted by $\mu^T$.

Note that the ordering of matrices is compatible with the matrix sum, product and transposition, i.e., for arbitrary matrices $\mu_1, \mu_2, \mu_1', \mu_2' \in S_{AXB}$ and $\eta_1, \eta_2 \in S_{BXC}$ we have that

$$\mu_1 \leq \mu_2 \text{ and } \mu_1' \leq \mu_2' \text{ implies } \mu_1 + \mu_1' \leq \mu_2 + \mu_2',$$

$$\mu_1 \leq \mu_2 \text{ and } \eta_1 \leq \eta_2 \text{ implies } \mu_1 \cdot \eta_1 \leq \mu_2 \cdot \eta_2,$$

$$\mu_1 \leq \mu_2 \text{ implies } \mu_1^T \leq \mu_2^T.$$

Matrices over the Boolean semiring are called Boolean matrices, and the set of all Boolean $A \times B$-matrices is denoted by $2^{AXB}$. If $S$ is an additively idempotent semiring, then $\{0, 1\}$ is a Boolean subsemiring of $S$, and matrices over $S$ taking values in $\{0, 1\}$ can be identified with Boolean matrices, i.e., we can treat $2^{AXB}$ as a subsemiring of $S_{AXB}$.

Moreover, Boolean matrices from $2^{AXB}$ can be identified with binary relations between the sets $A$ and $B$ so that the matrix sum and the Hadamard product correspond to the set-theoretical union and intersection, and the matrix product corresponds to the composition of binary relations. From this point of view, a Boolean matrix $\varrho \in 2^{AXB}$ is called reflexive, if $\varrho(a, a) = 1$, for every $a \in A$; symmetric, if $\varrho(a, b) = \varrho(b, a)$, for all $a, b \in A$; and transitive, if $\varrho(a, b) \cdot \varrho(b, c) = \varrho(a, c)$, for all $a, b, c \in A$. A reflexive and transitive Boolean matrix will be called a quasi-order matrix, and a reflexive, symmetric and transitive Boolean matrix will be called an equivalence matrix. It should be noted that every quasi-order matrix, and hence, every equivalence matrix, is idempotent, i.e., it satisfies $\varrho \cdot \varrho = \varrho$.

Let $\varrho \in 2^{AXA}$ be an equivalence matrix. For each $a \in A$, a vector $\varrho(a, \cdot)$ is called the equivalence class of $\varrho$ determined by $a$. For easier readability, we sometimes write $\varrho_a$ instead of $\varrho(a, \cdot)$. Classes of equivalence matrices have the same properties as classes of equivalence relations: $\varrho_0 = \varrho_e$ whenever $\varrho(a, b) = 1$, $\varrho_\emptyset \varrho_\emptyset = \varrho_\emptyset$ is the zero vector, whenever $\varrho(a, b) = 0$, and $\sum_{a \in A} \varrho_a$ is the vector whose all values are $1$. The set of all equivalence classes of $\varrho$ will be denoted by $A/\varrho$ and called the factor set of $A$ with respect to $\varrho$.

A Boolean matrix $\varrho \in 2^{AXB}$ is called a functional matrix if it corresponds to a binary relation which is a function, i.e., if for every $a \in A$ there exists $b \in B$ such that $\varrho(a, b) = 1$ (below, such matrices are called row complete), and $\varrho(a, b_1) = \varrho(a, b_2) = 1$ implies $b_1 = b_2$, for all $a \in A$ and $b_1, b_2 \in B$.

Let $A, B, C$ and $D$ be non-empty sets, and matrices $\mu \in S_{BXC}$, $\varrho_1 \in 2^{AXB}$ and $\varrho_2 \in 2^{CXD}$. Since $0$ and $1$ commute with all elements from $S$, we have

$$(\varrho_1 \cdot \mu)^T = \mu^T \cdot \varrho_1^T, \quad (\mu \cdot \varrho_2)^T = \varrho_2^T \cdot \mu^T.$$  \hspace{1cm} (4)

A Boolean matrix $\varrho \in 2^{AXB}$ is row complete if all its rows are non-zero vectors, i.e., if all its rows contain at least one $1$, and it is column complete if all its columns are non-zero vectors. Clearly, column complete functional matrices correspond to surjective functions. Note that $\varrho$ is row complete if and only if there is a function $\psi : A \rightarrow B$ such that $\varrho(a, \psi(a)) = 1$, for every $a \in A$. Let us call a function $\psi$ with this property a functional description of $\varrho$, and let us denote by $F(\varrho)$ the set of all such functions. For an equivalence matrix $\varrho \in 2^{B \times B}$, a function $\psi : A \rightarrow B$ is called $\varrho$-complete if for every $b \in B$ there exists $a \in A$ such that $\varrho(\psi(a), b) = 1$.

For a Boolean matrix $\varrho \in 2^{AXB}$ we define Boolean matrices $\varrho^A \in 2^{AXA}$ and $\varrho^B \in 2^{BXC}$ as follows for arbitrary $a_1, a_2 \in A$ and $b_1, b_2 \in B$:

$$\varrho^A(a_1, a_2) = [\varrho(a_1, \cdot) = \varrho(a_2, \cdot)], \quad \varrho^B(b_1, b_2) = [\varrho(\cdot, b_1) = \varrho(\cdot, b_2)].$$  \hspace{1cm} (5)

We call $\varrho^A$ the kernel, and $\varrho^B$ the co-kernel of $\varrho$. Clearly, both $\varrho^A$ and $\varrho^B$ are equivalence matrices.

A Boolean matrix $\varrho \in 2^{AXB}$ which satisfies $\varrho \cdot \varrho^T \cdot \varrho \leq \varrho$ is called a partial uniform matrix. Since the opposite inequality always holds, $\varrho$ is a partial uniform matrix if and only if $\varrho \cdot \varrho^T \cdot \varrho = \varrho$. A partial uniform matrix which is both row and column complete is called a uniform matrix. Note that every partial uniform matrix $\varrho$
can be converted into a uniform one by deleting all zero rows and columns in \( \varphi \). Just for this reason we use the name partial uniform matrix. For more information on uniform matrices we refer to [16, 17]. Here we mention only the following theorem from [16], as we shall use notation introduced in it.

**Theorem 2.2.** Let \( A \) and \( B \) be non-empty sets, and let \( \alpha \in 2^{A \times A} \) and \( \beta \in 2^{B \times B} \) be equivalence matrices. Then there exists a uniform matrix \( \varrho \in 2^{A \times B} \) such that \( \alpha = \varrho \gamma \) and \( \beta = \varrho \delta \) if and only if there exists a bijective function \( \phi : A/\alpha \to B/\beta \) that this function can be represented as \( \phi = \bar{\varrho} \), where \( \bar{\varrho} : A/\alpha \to B/\beta \) is a function given by
\[
\bar{\varrho}(a) = \beta_{\phi(a)}, \quad \text{for any } a \in A \text{ and } \psi \in F(\varrho).
\]

We also have that \( (\bar{\varrho})^{\top} = \varrho^{\top} \).

3. Weighted automata

In the rest of the paper, let \( X \) a finite non-empty alphabet.

A weighted finite automaton, or just a weighted automaton, over \( X \) and \( S \) is a quadruple \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \), where \( A \) is a finite non-empty set of states, \( \delta^A : A \times X \times A \to S \) is a weighted transition function, \( \sigma^A \in S^A \) is an initial weight vector and \( \tau^A \in S^A \) is a final weight vector. For each \( x \in X \) we define a weighted transition matrix \( \delta^A_x \in S^{A \times A} \) by letting \( \delta^A_x(a, b) = \delta^A(a, x, b) \), for all \( a, b \in A \). Also, for any \( u \in X^* \) we define a matrix \( \delta^A_u \in S^{A \times A} \) inductively, as follows: \( \delta^A_1 = \delta^A \) is the unit matrix, and for all \( u \in X^* \) and \( x \in X \) we set \( \delta^A_x = \delta^A_x \cdot \delta^A_u \).

A formal power series over \( X \) and \( S \), for short a series, is any mapping \( \varphi : X^* \to S \). Instead of \( \varphi(u) \) we write \( \langle \varphi, u \rangle \) for every \( u \in X^* \). The set of all series over \( X \) and \( S \) is denoted by \( S\langle X^* \rangle \). The behavior of a weighted automaton \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) is the series \( \langle \mathcal{A} \rangle \) in \( S\langle X^* \rangle \) defined by
\[
\langle \mathcal{A} \rangle, u) = \alpha' \cdot \delta^A_{u_1} \cdot \tau^A = \sum_{a_1, a_2 \in A} \alpha'(a_1) \cdot \delta^A_{u_1}(a_1, a_2) \cdot \tau^A(a_2),
\]
for each \( u \in X^* \).

Weighted automata \( \mathcal{A} \) and \( \mathcal{B} \) are said to be equivalent if they have the same behavior, i.e., if \( \langle \mathcal{A} \rangle = \langle \mathcal{B} \rangle \). Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) be a weighted automaton and let \( \varphi \in 2^{A \times A} \) be an equivalence matrix. Without any restriction on \( \varphi \) we can define a weighted transition function \( \delta^{A/\varphi} : (A/\varphi) \times X \times (A/\varphi) \to S \) by
\[
\delta^{A/\varphi}_x(a, _b) = \varphi_a \cdot \delta^A_x(a, b) = \sum_{a', b' \in A} \varphi(a, a') \cdot \delta^A_x(a', x, b') \cdot \varphi(b', b)
\]
for arbitrary \( a, b \in A \) and \( x \in X \). We can also define an initial weight vector \( \sigma^{A/\varphi} \in S^{A/\varphi} \) and a final weight vector \( \tau^{A/\varphi} \in S^{A/\varphi} \) as follows for every \( a \in A \):
\[
\sigma^{A/\varphi}(a) = \sigma^A \cdot \varphi(a) = \sum_{a' \in A} \sigma^A(a') \cdot \varphi(a'),
\]
\[
\tau^{A/\varphi}(a) = \tau^A \cdot \varphi(a) = \sum_{a' \in A} \tau^A(a') \cdot \varphi(a'),
\]
Clearly, \( \delta^{A/\varphi}, \sigma^{A/\varphi} \) and \( \tau^{A/\varphi} \) are well-defined and \( \mathcal{A}/\varphi = (A/\varphi, \delta^{A/\varphi}, \sigma^{A/\varphi}, \tau^{A/\varphi}) \) is a weighted automaton, called the factor weighted automaton of \( \mathcal{A} \) with respect to \( \varphi \).

The series \( \langle \mathcal{A}/\varphi \rangle \) of the factor weighted automaton \( \mathcal{A}/\varphi \) is given by
\[
\langle \mathcal{A}/\varphi \rangle, u) = \sigma^A \cdot \varphi \cdot \tau^A,
\]
\[
\langle \mathcal{A}/\varphi \rangle, u) = \sigma^A \cdot \varphi \cdot \delta^A_{u_1} \cdot \varphi \cdot \delta^A_{u_2} \cdot \varphi \cdots \varphi \cdot \delta^A_{u_n} \cdot \varphi \cdot \tau^A,
\]
for any \( u = x_1x_2 \cdots x_n \in X^n \), where \( x_1, x_2, \ldots, x_n \in X \). Hence, the weighted automata \( \mathcal{A} \) and \( \mathcal{A}/\varphi \) are equivalent if and only if the following system of matrix equations
\[
\sigma^A \cdot \tau^A = \sigma^A \cdot \varphi \cdot \tau^A,
\]
\[
\sigma^A \cdot \delta^A_{u_1} \cdot \delta^A_{u_2} \cdot \delta^A_{u_3} \cdots \delta^A_{u_n} \cdot \tau^A = \sigma^A \cdot \varphi \cdot \delta^A_{u_1} \cdot \varphi \cdot \delta^A_{u_2} \cdot \varphi \cdots \varphi \cdot \delta^A_{u_n} \cdot \varphi \cdot \tau^A,
\]

6
for all $n \in \mathbb{N}$ and $x_1, x_2, \ldots, x_n \in X$. We call (13) the general system. The general system has at least one solution, the unit $A \times A$-matrix, which is called the trivial solution. However, the general system consists of infinitely many equations, and finding its nontrivial solutions may be a very difficult task. For that reason we will consider some instances of the general system, by which we mean systems, built from the same matrices, whose sets of solutions are contained in the set of all solutions to the general system.

Let $\mathcal{A} = (A, \delta^A, \sigma^A, \tau^A)$ and $\mathcal{B} = (B, \delta^B, \sigma^B, \tau^B)$ be weighted automata. The automaton $\mathcal{B}$ is said to be a subautomaton of $\mathcal{A}$ if $B \subseteq A$, $\delta^B$ is the restriction of $\delta^A$ to $B \times X \times B$, and $\sigma^B$ and $\tau^B$ are restrictions of $\sigma^A$ and $\tau^A$ to $B$.

A function $\phi : A \rightarrow B$ is called an isomorphism between $\mathcal{A}$ and $\mathcal{B}$ if it is bijective and the following is true for all $a, a_1, a_2 \in A$ and $x \in X$:

$$\delta^A_\phi(a_1, a_2) = \delta^B_\phi(\phi(a_1), \phi(a_2)), \quad (14)$$
$$\sigma^A_\phi(a) = \sigma^B_\phi(\phi(a)), \quad (15)$$
$$\tau^A_\phi(a) = \tau^B_\phi(\phi(a)). \quad (16)$$

If there exists an isomorphism between $\mathcal{A}$ and $\mathcal{B}$, then we say that $\mathcal{A}$ and $\mathcal{B}$ are isomorphic weighted automata, and we write $\mathcal{A} \cong \mathcal{B}$. In other words, two weighted automata are isomorphic if in essence they have the same structure. Clearly, the inverse of an isomorphism of weighted automata is also an isomorphism, as well as the composition of two isomorphisms.

A function $\phi : A \rightarrow B$ which is injective and satisfies (14)–(16) is called a monomorphism from $\mathcal{A}$ into $\mathcal{B}$. It is easy to check that $\phi : A \rightarrow B$ is a monomorphism from $\mathcal{A}$ to $\mathcal{B}$ if and only if it is an isomorphism from $\mathcal{A}$ to the subautomaton $\mathcal{C} = (C, \delta^C, \sigma^C, \tau^C)$ of $\mathcal{B}$, where $C = \text{Im} \phi$.

The reverse weighted automaton of a weighted automaton $\mathcal{A} = (A, \delta^A, \sigma^A, \tau^A)$ is a weighted automaton $\mathcal{A}^r = (A, \delta^A, \tau^A, \sigma^A)$, where $\delta^A = (\delta^A)^T$, for each $x \in X$.

**Proposition 3.1.** Let $\mathcal{A} = (A, \delta^A, \sigma^A, \tau^A)$ be a weighted automaton, and let $\alpha, \beta \in 2^{A \times A}$ be equivalence matrices such that $\alpha \leq \beta$. Then the Boolean matrix $\beta_a / \alpha \in 2^{(A/\alpha) \times (A/\alpha)}$ defined by

$$\beta_a(a_1, a_2) = \beta(a_1, a_2), \quad \text{for all } a_1, a_2 \in A,$$

is an equivalence matrix on $\mathcal{A} / \alpha$ and the factor weighted automata $(\mathcal{A} / \alpha)(\beta_a / \alpha)$ and $\mathcal{A} / \beta$ are isomorphic.

### 4. Bismulations for weighted automata

In this section we give definitions and discuss the basic properties of simulations and bisimulations between weighted automata.

#### 4.1. Simulations and bisimulations between two weighted automata

Let $\mathcal{A} = (A, \delta^A, \sigma^A, \tau^A)$ and $\mathcal{B} = (B, \delta^B, \sigma^B, \tau^B)$ be weighted automata. A Boolean matrix $\varrho \in 2^{A \times B}$ is called a forward simulation between $\mathcal{A}$ and $\mathcal{B}$ if

$$\sigma^A \leq \sigma^B \cdot \varrho^T, \quad (fs-1)$$
$$\varrho^T \cdot \delta^A \leq \delta^B \cdot \varrho^T, \quad \text{for every } x \in X, \quad (fs-2)$$
$$\varrho^T \cdot \tau^A \leq \tau^B. \quad (fs-3)$$

Note that if (fs-2) holds for every letter $x \in X$, then it also holds if we replace the letter $x$ by an arbitrary word $u \in X'$. We call $\varrho$ a backward simulation between $\mathcal{A}$ and $\mathcal{B}$ if it is a forward simulation between the reverse automata $\mathcal{A}^r$ and $\mathcal{B}^r$. Furthermore, $\varrho$ is called forward bisimulation if both $\varrho$ and $\varrho^T$ are forward simulations, and a backward bisimulation, if both $\varrho$ and $\varrho^T$ are backward simulations. If $\varrho$ is a forward simulation and $\varrho^T$ is a backward simulation, then $\varrho$ is called a forward-backward bisimulation, and if $\varrho$ is a backward simulation and $\varrho^T$ is a forward simulation, then $\varrho$ is called a backward-forward bisimulation. For the sake of simplicity, we will call $\varrho$ just a simulation if $\varrho$ is either a forward or a backward simulation, and just a bisimulation if
\(q\) is any of the four types of bisimulations defined above. Moreover, forward and backward bisimulations will be called *homotypic*, whereas backward-forward and forward-backward bisimulations will be called *heterotypic*.

The meaning of forward and backward simulations can be best explained in the case when \(\mathcal{A}\) and \(\mathcal{B}\) are weighted automata over the Boolean semiring (i.e., nondeterministic automata). For this purpose we will use the diagram shown in Figure 1. Let \(q\) be a forward simulation between \(\mathcal{A}\) and \(\mathcal{B}\) and let \(a_0, a_1, \ldots, a_n\) be an arbitrary successful run of the automaton \(\mathcal{A}\) on a word \(u = x_1 x_2 \cdots x_n\) \((x_1, x_2, \ldots, x_n \in X)\), i.e., a sequence of states of \(\mathcal{A}\) such that \(a_0 \in \sigma^A\), \((a_k, a_{k+1}) \in \delta^A_{\sigma^A}\), for \(0 \leq k \leq n - 1\), and \(a_n \in \tau^A\). According to (fs-1), there is an initial state \(b_0 \in \sigma^B\) such that \((a_0, b_0) \in q\). Suppose that for some \(k, 0 \leq k \leq n - 1\), we have built a sequence of states \(b_0, b_1, \ldots, b_k\) such that \((b_{i-1}, b_i) \in \delta^B_{\sigma^B}\) and \((a_i, b_i) \in q\), for each \(i, 1 \leq i \leq k\). Then \((b_k, a_{k+1}) \in \eta^B \cdot \delta^A_{\sigma^A}\) and by (fs-2) we obtain that \((b_k, a_{k+1}) \in \delta^B_{\sigma^B} \cdot \eta^B\), so there exists \(b_{k+1} \in B\) such that \((b_k, b_{k+1}) \in \delta^B_{\sigma^B}\), and \((a_{k+1}, b_{k+1}) \in q\). Therefore, we have successively built a sequence \(b_0, b_1, \ldots, b_n\) of states of \(\mathcal{B}\) such that \(b_0 \in \sigma^B\), \((b_k, b_{k+1}) \in \delta^B_{\sigma^B}\), for every \(k, 0 \leq k \leq n - 1\), and \((a_k, b_k) \in q\), for each \(k, 0 \leq k \leq n\). Moreover, by (fs-3) we obtain that \(b_n \in \tau^B\). Thus, the sequence \(b_0, b_1, \ldots, b_n\) is a successful run of the automaton \(\mathcal{B}\) on the word \(u\) which simulates the original run \(a_0, a_1, \ldots, a_n\) of \(\mathcal{A}\) on \(u\). In contrast to forward simulations, where we build the sequence \(b_0, b_1, \ldots, b_n\) moving forward, starting with \(b_0\) and ending with \(b_n\), in the case of backward simulations we build this sequence moving backward, starting with \(b_n\) and ending with \(b_0\). In a similar way we can understand forward and backward simulations between arbitrary weighted automata, taking into account transition weights.

The following theorem presents some of the most important properties of simulations and bisimulations.

**Theorem 4.1.** Let \(\mathcal{A} = (A, \delta^A, \sigma^A, \tau^A)\) and \(\mathcal{B} = (B, \delta^B, \sigma^B, \tau^B)\) be weighted automata and let \(q \in 2^{AXB}\) be a Boolean matrix. Then

(A) If \(q\) is a simulation, then \(\llbracket \mathcal{A} \rrbracket \leq \llbracket \mathcal{B} \rrbracket\).

(B) If \(q\) is a bisimulation, then \(\llbracket \mathcal{A} \rrbracket = \llbracket \mathcal{B} \rrbracket\).

**Proof.** (A) Let \(q\) be a forward simulation. Then for arbitrary \(u \in X^*\) we have that

\[
\llbracket \mathcal{A} \rrbracket, u) = \sigma^A \cdot \delta^A \cdot \tau^A \leq \sigma^B \cdot \eta^B \cdot \delta^A \cdot \tau^A \leq \sigma^B \cdot \delta^B \cdot \tau^B = \llbracket \mathcal{B} \rrbracket, u).
\]

Therefore, \(\llbracket \mathcal{A} \rrbracket, u) \leq \llbracket \mathcal{B} \rrbracket, u)\). Similarly we prove the case when \(q\) is a backward simulation.

(B) This statement is a direct consequence of the definition of the four types of bisimulations and statement (A).

Note that for any statement on forward simulations or bisimulations which is universally valid there is the corresponding universally valid statement on backward simulations or bisimulations. For that reason, here we deal only with forward simulations and bisimulations.
The following theorem presents some fundamental properties of simulations and bisimulations.

**Theorem 4.2.** If there exists at least one forward simulation (resp. bisimulation) between weighted automata \( \mathcal{A} \) and \( \mathcal{B} \), then there exists the greatest forward simulation (resp. bisimulation) between \( \mathcal{A} \) and \( \mathcal{B} \).

Moreover, the greatest forward bisimulation, if it exists, is a partial uniform Boolean matrix.

**Proof.** It is easy to verify that the sum of an arbitrary non-empty family of forward simulations between \( \mathcal{A} \) and \( \mathcal{B} \) is also a forward simulation. Therefore, if there exists at least one forward simulation between \( \mathcal{A} \) and \( \mathcal{B} \), then the sum of all forward simulations between \( \mathcal{A} \) and \( \mathcal{B} \) is the greatest forward simulation between \( \mathcal{A} \) and \( \mathcal{B} \). The same arguments are valid for forward bisimulations.

It is also easy to see that the composition of two forward bisimulations is also a forward bisimulation. Thus, if \( q \) is the greatest forward bisimulation between \( \mathcal{A} \) and \( \mathcal{B} \), then \( q \cdot q^\top \cdot q \) is also a forward bisimulation between \( \mathcal{A} \) and \( \mathcal{B} \), whence \( q \cdot q^\top \cdot q \leq q \). This means that \( q \) is a partial uniform Boolean matrix.

The previous theorem points to the importance of studying those forward bisimulations which are partially uniform Boolean matrices. However, bisimulations are intended to model the equivalence between automata (or some related systems), and two automata are considered to be equivalent (as a whole) if every state of the first automaton is equivalent to some state of the second automaton, and vice versa. In the context of weighted automata this means that a Boolean matrix which we use to model the equivalence between weighted automata (or some related systems), and two automata are considered to be equivalent (as a whole) if every state of the first automaton is equivalent to some state of the second automaton, and vice versa. In the context of weighted automata this means that a Boolean matrix which we use to model the equivalence between weighted automata (or some related systems), and two automata are considered to be equivalent (as a whole) if every state of the first automaton is equivalent to some state of the second automaton, and vice versa. In the context of weighted automata this means that a Boolean matrix which we use to model the equivalence between weighted automata (or some related systems), and two automata are considered to be equivalent (as a whole) if every state of the first automaton is equivalent to some state of the second automaton, and vice versa.

**4.2. Simulations and bisimulations on a single weighted automaton**

Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) be a weighted automaton. By a simulation/bisimulation on \( \mathcal{A} \) of a given type we mean any simulation/bisimulation of this type between \( \mathcal{A} \) and itself. Since the unit \( A \times A \)-matrix is a forward bisimulation on \( \mathcal{A} \), and the composition of any two forward simulations on \( \mathcal{A} \) is also a forward simulation, we obtain the following proposition.

**Proposition 4.3.** Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) be a weighted automaton. Then the greatest forward simulation on \( \mathcal{A} \) is a quasi-order matrix, and the greatest forward bisimulation on \( \mathcal{A} \) is an equivalence matrix.

A forward bisimulation on a weighted automaton \( \mathcal{A} \) which is an equivalence matrix will be called a forward bisimulation equivalence matrix on \( \mathcal{A} \). It should be noted that an equivalence matrix \( q \in 2^{A \times A} \) is a forward bisimulation on a weighted automaton \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) if and only if it satisfies

\[
q \cdot \delta^A \leq \delta^A \cdot q, \quad \text{for every } x \in X,  
\]

\[
q \cdot \tau^A \leq \tau^A, \quad \text{(condition } \sigma^A \leq \sigma^A \cdot q \text{ is satisfied whenever } q \text{ is a reflexive Boolean matrix). Note also that conditions (18) and (19) can be written in the equivalent form}  
\]

\[
q \cdot \delta^A \cdot q = \delta^A \cdot q, \quad \text{for every } x \in X,  
\]

\[
q \cdot \tau^A = \tau^A. \quad \text{(21)}
\]

The following theorem highlights the role of forward bisimulation equivalence matrices in the state reduction of weighted automata.

**Theorem 4.4.** Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) be a weighted automaton, \( q \) a forward bisimulation equivalence matrix, and \( \mathcal{A}/q = (A/q, \delta^{A/q}, \sigma^{A/q}, \tau^{A/q}) \) the factor weighted automaton of \( \mathcal{A} \) with respect to \( q \).

Then \( \mathcal{A} \) and \( \mathcal{A}/q \) are equivalent weighted automata.

**Proof.** It is easy to see that the system consisting of equations (20) and (21) is instance of the general system (13), i.e., every solution to (20) and (21) is a solution to the general system. Therefore, \( q \) is a solution to the general system, which means that weighted automata \( \mathcal{A} \) and \( \mathcal{A}/q \) are equivalent.
5. Construction of the greatest simulations and bisimulations

In this section we give algorithms which decide whether there are simulations/bisimulations of a given type between weighted automata, and whenever this is the case, the algorithms provide constructions of the greatest simulations/bisimulations. These algorithms are based on the concepts of relative (Boolean) residuals of matrices, which we introduce in the sequel.

Consider non-empty sets $A$, $B$ and $C$. Let $\alpha \in S^{C \times A}$ and $\beta \in S^{C \times B}$. By a right residual of $\beta$ by $\alpha$ we mean the greatest solution (if it exists) of the matrix inequality

$$\alpha \cdot \chi \leq \beta,$$  \hspace{1cm} (22)

where $\chi$ is an unknown matrix taking values in $S^{A \times B}$. Dually, for $\alpha \in S^{A \times C}$ and $\beta \in S^{B \times C}$, by a left residual of $\beta$ by $\alpha$ we mean the greatest solution (if it exists) of the matrix inequality

$$\chi \cdot \alpha \leq \beta,$$  \hspace{1cm} (23)

where $\chi$ is an unknown matrix taking values in $S^{B \times A}$.

For some semirings, right and left residuals exist for each pair of matching matrices, but in general, the residuals do not necessarily exist. The question naturally arises: If there is no greatest solution to the residuals of matrices, which we introduce in the sequel.

Consider an arbitrary $\xi \in S^{A \times B}$.

Proof. Consider an arbitrary $\xi \in S^{A \times B}$. Suppose that $\alpha \cdot \xi \leq \beta$, and let $(a, b) \in A \times B$ such that $\xi(a, b) = 1$. Then for every $c \in C$ we have that

$$\alpha(c, a) = \alpha(c, a) \cdot \xi(a, b) \leq \sum_{a' \in A} \alpha(c, a') \cdot \xi(a', b) = (\alpha \cdot \xi)(c, b) \leq \beta(c, b),$$

and thus, $(\alpha \cdot \xi)(a, b) = 1$. Consequently, $\xi \leq \alpha \cdot \beta$.

Conversely, if $\xi \leq \alpha \cdot \beta$, then for an arbitrary $(c, b) \in C \times B$ we have that

$$(\alpha \cdot \xi)(c, b) = \sum_{a \in A} \alpha(c, a) \cdot \xi(a, b) \leq \sum_{a \in A} \alpha(c, a) \cdot (\alpha \cdot \beta)(a, b) = \sum_{a \in A} \alpha(c, a) \leq n \cdot (\beta(c, b)) \leq \beta(c, b),$$

where $A' = \{ a \in A \mid (\alpha \cdot \beta)(a, b) = 1 \} = \{ a \in A \mid \alpha(a, b) \leq \beta \}$ and $n = |A'|$. Therefore, $\alpha \cdot \xi \leq \beta$, and we have proved the adjunction property (25).

According to (25), $\alpha \cdot \beta$ is the greatest solution to (22) in $S^{A \times B}$, i.e., it is the Boolean residual of $\beta$ by $\alpha$. \hfill \Box

Moreover, the following dual theorem holds:
Theorem 5.2. For any \( \alpha \in S^{A \times C} \) and \( \beta \in S^{B \times C} \) let a Boolean matrix \( \beta/\alpha \) be defined by
\[
(\beta/\alpha)(b, a) = [\alpha(a, \cdot) \leq \beta(b, \cdot)],
\]
for all \( a \in A \) and \( b \in B \). Then for each \( \xi \in 2^{B \times A} \) the following adjunction property holds
\[
\xi \cdot \alpha \leq \beta \iff \xi \leq \beta/\alpha,
\]
and \( \beta/\alpha \) is the Boolean left residual of \( \beta \) by \( \alpha \).

Note that for vectors \( \nu \in S^A \) and \( \eta \in S^B \) (treated as \( C \times A \) and \( C \times B \) matrices, or \( A \times C \) and \( B \times C \) matrices, where \( |C| = 1 \), the Boolean right residual \( \nu \setminus \eta \in 2^{A \times B} \) and the Boolean left residual \( \eta/\nu \in 2^{B \times A} \) are given by
\[
(\nu \setminus \eta)(a, b) = (\eta/\nu)(b, a) = [\nu(a) \leq \eta(b)],
\]
for all \( a \in A \) and \( b \in B \), i.e., \( \nu \setminus \eta = (\eta/\nu)^\top \).

The next theorem provides equivalent forms of conditions (fs-2) and (fs-3) in the definition of a forward simulation.

Theorem 5.3. Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) and \( \mathcal{B} = (B, \delta^B, \sigma^B, \tau^B) \) be weighted automata. A Boolean matrix \( q \in 2^{A \times B} \) satisfies conditions (fs-2) and (fs-3) if and only if it satisfies
\[
q \leq \bigotimes_{x \in X} \left[ (\delta^B_x \cdot q^\top) / \delta^A_x \right]^\top, \quad q \leq \tau^A \setminus \tau^B.
\]

Proof. Consider an arbitrary Boolean matrix \( q \in 2^{A \times B} \). According to Theorem 5.1, the matrix \( q \) satisfies condition (fs-3) if and only if \( q^\top \leq \tau^B / \tau^A = (\tau^A \setminus \tau^B)^\top \), which is equivalent to \( q \leq \tau^A \setminus \tau^B \).

On the other hand, according to Theorem 5.1, \( q \) satisfies (fs-2) if and only if
\[
q^\top \leq (\delta^B_x \cdot q^\top) / \delta^A_x, \quad \text{for every } x \in X,
\]
which is equivalent to
\[
q \leq \bigotimes_{x \in X} \left[ (\delta^B_x \cdot q^\top) / \delta^A_x \right]^\top.
\]

Therefore, a Boolean matrix \( q \in 2^{A \times B} \) satisfies (fs-2) and (fs-3) if and only if it satisfies (28). \( \square \)

Now we are ready to prove a theorem which provides a method for testing the existence of a forward simulation between two weighted automata and the construction of the greatest forward simulation, if forward simulations exist.

Theorem 5.4. Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) and \( \mathcal{B} = (B, \delta^B, \sigma^B, \tau^B) \) be weighted automata, and let \( \{q_k\}_{k \in \mathbb{N}} \subseteq 2^{A \times B} \) be a sequence of Boolean matrices defined inductively by
\[
q_1 = \tau^A \setminus \tau^B, \quad q_{k+1} = q_k \oplus \left( \bigotimes_{x \in X} \left[ (\delta^B_x \cdot q_k^\top) / \delta^A_x \right]^\top \right), \quad \text{for every } k \in \mathbb{N}.
\]

Then the following holds:

(a) The sequence \( \{q_k\}_{k \in \mathbb{N}} \) is finite and descending, and there is the least natural number \( m \in \mathbb{N} \) such that \( q_m = q_{m+1} \);
(b) \( q_m \) is the greatest Boolean matrix in \( 2^{A \times B} \) which satisfies (fs-2) and (fs-3);
(c) If \( q_m \) satisfies (fs-1), then it is the greatest forward simulation between \( \mathcal{A} \) and \( \mathcal{B} \);
(d) If \( q_m \) does not satisfy (fs-1), then there is no forward simulation between \( \mathcal{A} \) and \( \mathcal{B} \).
Proof. (a) Clearly, the sequence \( \{\delta_k\}_{k \in \mathbb{N}} \) is descending. Since the sets \( A \) and \( B \) are finite, we have that \( 2^A \times B \) is finite, so the sequence \( \{\delta_k\}_{k \in \mathbb{N}} \) is also finite. Thus, there exists the least natural number \( m \) such that \( \theta_m = \theta_{m+1} \).

(b) By \( \theta_m = \theta_{m+1} \) it follows that

\[
\theta_m \leq \bigcap_{x \in X} \left( (\delta_x^B \cdot \delta_m) / \delta_A^x \right),
\]

and also, \( \theta_m \leq \theta_1 = \tau_A \setminus \tau_B \). Therefore, \( \theta_m \) satisfies \( (fs-2) \) and \( (fs-3) \).

Let \( \varrho \in 2^A \times B \) be an arbitrary Boolean matrix which satisfies \( (fs-2) \) and \( (fs-3) \). As we have already said, \( \varrho \) satisfies condition \( (fs-3) \) if and only if \( \varrho \leq \tau_A \setminus \tau_B = \varrho_1 \). Next, suppose that \( \varrho \leq \varrho_k \) for some \( k \in \mathbb{N} \). Then for every \( x \in X \) we have that \( \varrho^T \cdot \delta_x^A \leq \delta_x^B \cdot \varrho^T \leq \delta_x^B \cdot \delta_x^k \), and according to Theorem \( 5.1 \), \( \varrho^T \leq (\delta_x^B \cdot \delta_x^k) / \delta_A^x \), or equivalently, \( \varrho \leq [(\delta_x^B \cdot \delta_x^k) / \delta_A^x] \). Therefore,

\[
\varrho \leq \varrho \cap \left( \bigcap_{x \in X} \left( (\delta_x^B \cdot \delta_x^k) / \delta_A^x \right) \right) = \varrho_{k+1}.
\]

Now, by induction we conclude that \( \varrho \leq \varrho_k \), for each \( k \in \mathbb{N} \), and hence \( \varrho \leq \varrho_m \). Thus, \( \varrho_m \) is the greatest Boolean matrix in \( 2^{A \times B} \) which satisfies \( (fs-2) \) and \( (fs-3) \).

(c) This follows immediately from assertion (b).

(d) Suppose that \( \varrho_m \) does not satisfy \( (fs-1) \). Let \( \varrho \in 2^{A \times B} \) be an arbitrary forward simulation between \( \mathcal{A} \) and \( \mathcal{B} \). According to (b) of this theorem, \( \varrho \leq \varrho_m \), so \( \sigma^A \leq \sigma^B \cdot \varrho^T \leq \sigma^B \cdot \varrho_m \). But this is in contradiction with our starting assumption that \( \varrho_m \) does not satisfy \( (fs-1) \). Hence, we conclude that no Boolean matrix in \( 2^{A \times B} \) satisfies \( (fs-1) \), \( (fs-2) \) and \( (fs-3) \). \( \square \)

The previous theorem can be transformed into the following algorithm.

**Algorithm 5.5** (Computation of the greatest forward simulation). The input of this algorithm are weighted automata \( \mathcal{A} = (A, \delta_A, \sigma_A, \tau_A) \) and \( \mathcal{B} = (B, \delta_B, \sigma_B, \tau_B) \). The algorithm decides whether there is a forward simulation between \( \mathcal{A} \) and \( \mathcal{B} \), and when it exists, the output of the algorithm is the greatest forward simulation.

The procedure is to construct the sequence of Boolean matrices \( \{\varrho_k\}_{k \in \mathbb{N}} \) in the following way:

(A1) In the first step we compute \( \tau_A \setminus \tau_B \) and we set \( \varrho_1 = \tau_A \setminus \tau_B \).

(A2) After the \( k \)th step let \( \varrho_k \) be the Boolean matrix that has been constructed.

(A3) In the next step we construct the Boolean matrix \( \varrho_{k+1} \) by means of the formula (29).

(A4) Simultaneously, we check whether \( \varrho_{k+1} = \varrho_k \).

(A5) When we find the smallest number \( m \) such that \( \varrho_{m+1} = \varrho_m \), the procedure of constructing the sequence \( \{\varrho_k\}_{k \in \mathbb{N}} \) terminates, and we check whether \( \varrho_m \) satisfies \( (fs-1) \).

If \( \varrho_m \) satisfies \( (fs-1) \), then it is the greatest forward simulation between \( \mathcal{A} \) and \( \mathcal{B} \), and if \( \varrho_m \) does not satisfy \( (fs-1) \), then there is no any forward simulation between \( \mathcal{A} \) and \( \mathcal{B} \).

Since \( 2^{A \times B} \) is finite, the sequence \( \{\varrho_k\}_{k \in \mathbb{N}} \) is also finite, and the algorithm terminates in a finite number of steps, for any weighted automaton over an additively idempotent semiring.

Let us discuss the computational time of this algorithm. In (A1) we compute the Boolean right residual of vectors \( \tau_A \) and \( \tau_B \) in \( 2^A \) and \( 2^B \), where we have to perform \( |A||B| \) inequality checks. Since the ordering in \( S \) is defined according to rule (1), the computational time of any single inequality check can be given as \( c_s \), the computational time of the addition in \( S \). Therefore, the computational time of (A1) is \( O(|A||B|c_s) \).

In (A3) we first compute the product of the matrix \( \delta_B^c \in S^{B \times B} \) and the Boolean matrix \( \varrho_k^c \in 2^{B \times A} \), and if this computation is performed according to the definition of matrix product, then its computational time is \( O(|A||B|^2c_s) \). After that, we compute the Boolean left residual of matrices \( \delta_B^c \cdot \varrho_k^c \in S^{B \times A} \) and \( \delta_A^k \in S^{A \times A} \), where we have to perform \( |A|^2|B| \) inequality checks, so the computational time of this part is \( O(|A|^2|B|c_s) \). Hence, we compute any single \( (\delta_B^c \cdot \varrho_k^c / \delta_A^k)^\top \) in time \( O(|A||B|(|A|+|B|)c_s) \), and the whole collection \( \{(\delta_B^c \cdot \varrho_k^c / \delta_A^k)^\top\}_{x \in X} \) in time \( O(|A||B|^2c_s) \).
in time \(O(\|A\|B(\|A\|+\|B\|)\|X\|c_\ast)\). Finally, we compute \(q_{k+1}\) making \(|X|\) Hadamard products of \(A \times B\)-matrices, what can be done in time \(O(\|A\|\|B\|\|X\|)\). Thus, the computational time of (A3) is \(O(\|A\|\|B\|\|A\|+\|B\|\|X\|c_\ast)\).

In (A4) computational time to check whether \(q_{k+1} = q_k\) is \(O(\|A\|\|B\|\|X\|)\).

Therefore, the total computational time of any single iteration is \(O(\|A\|\|B\|\|A\|+\|B\|\|X\|c_\ast)\) (except the first one, which is faster). It remains to estimate the number of iterations.

For a Boolean vector \(v \in 2^A\) let \(\text{size}(v)\) be the number of entries in \(v\) which are equal to 1. Since \(\{q_k\}_{k \in \mathbb{N}}\) is a descending sequence of matrices, then for any \(a \in A\), the sequence of vectors \(\{q_k(a, \cdot)\}_{k \in \mathbb{N}}\) and the sequence of numbers \(\{\text{size}(q_k(a, \cdot))\}_{k \in \mathbb{N}}\) are also descending. We also have that \(0 \leq \text{size}(q_k(a, \cdot)) \leq \|B\|\), for each \(k \in \mathbb{N}\), and consequently, \(q_k(a, \cdot) = q_k(a, \cdot)\), for \(k = \|B\| + 1\), every \(l \in \mathbb{N}\) and every \(a \in A\). Therefore, \(q_k = q_{k+1}\), for \(k = \|B\| + 1\) and every \(l \in \mathbb{N}\), which means that the number of iterations is less or equal \(|B| + 1\). In the same way we conclude that the number of iterations is less or equal \(|A| + 1\), and we find that the algorithm terminates after at most \(m\) iterations, where \(m = \min(|A| + 1, |B| + 1)\).

Finally, the time required to check whether \(q_m\) satisfies \((fs-1)\) is \(O(\|A\|\|B\|c_\ast)\).

Summing up, we get that the total computation time for the whole algorithm is \(O(m\|A\|\|B\|\|A\|+\|B\|\|X\|c_\ast)\), and hence, the algorithm is polynomial-time.

Similarly we can give a procedure which decides whether there exists a forward (or backward-forward) bisimulation between \(A\) and \(B\), and whenever there is at least one such bisimulation, the algorithm computes the greatest one. The only difference is that for forward bisimulations we build the sequence \(\{q_k\}_{k \in \mathbb{N}}\) by

\[
q_1 = (\tau^A \setminus \tau^B) \odot (\tau^A / \tau^B), \quad q_{k+1} = q_k \odot \left( \bigoplus_{x \in X} \left( [\tau^A_x \cdot q_k] / [\tau^B_x] \right) \right) \odot \left( [\tau^A_x / q_k] \cdot [\tau^B_x] \right), \tag{30}
\]

and at the final stage of the algorithm, we perform the check using conditions \(\sigma^A \leq \sigma^B \cdot q^\top\) and \(\sigma^B \leq \sigma^A \cdot q\) instead of \((fs-1)\), and in the case of backward-forward bisimulations we build \(\{q_k\}_{k \in \mathbb{N}}\) by

\[
q_1 = (\sigma^A \setminus \sigma^B) \odot (\tau^A / \tau^B), \quad q_{k+1} = q_k \odot \left( \bigoplus_{x \in X} \left( [\tau^A_x / q_k] / [\tau^B_x] \right) \right) \odot \left( [\tau^A_x \cdot q_k] \cdot [\tau^B_x] \right), \tag{31}
\]

and in the check at the final stage of the algorithm we use conditions \(\sigma^B \leq \sigma^A \cdot q\) and \(\tau^A \leq q \cdot \tau^B\). Let us note that in the case of simulations and bisimulations on a single weighted automaton we do not need to check whether \(q_m\) satisfies condition analogous to \((fs-1)\). This condition is automatically satisfied, since every Boolean matrix in the sequence \(\{q_k\}_{k \in \mathbb{N}}\), and hence the matrix \(q_m\), is reflexive. Therefore, when we find the smallest number \(m\) such that \(q_{m+1} = q_m\), then \(q_m\) is the required simulation/bisimulation.

The following example demonstrates the work of the above algorithm.

Example 5.6. Let \(A = (A^A, \sigma^A, \tau^A)\) and \(B = (B^B, \sigma^B, \tau^B)\) be weighted automata over an alphabet \(X = \{x, y\}\) and the max-plus semiring \((\mathbb{N} \cup \{-\infty\}, \max, +, -\infty, 0)\), with \(|A| = 3\) and \(|B| = 2\), which are represented by the following graph:

![Graph 1](image1.jpg)

![Graph 2](image2.jpg)
They can also be represented by the following matrices and vectors:

\[ \sigma^A = \begin{bmatrix} 5 & 0 & 0 \end{bmatrix}, \quad \delta^A_x = \begin{bmatrix} 10 & 3 & 4 \\ 5 & 10 & 3 \\ 4 & 6 & 7 \end{bmatrix}, \quad \delta^A_y = \begin{bmatrix} 5 & 6 & 2 \\ 3 & 3 & 4 \\ 7 & 7 & 10 \end{bmatrix}, \quad \tau^A = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \]

\[ \sigma^B = \begin{bmatrix} 5 & 0 \end{bmatrix}, \quad \delta^B_x = \begin{bmatrix} 10 & 6 \\ 5 & 7 \end{bmatrix}, \quad \delta^B_y = \begin{bmatrix} 6 & 6 \\ 7 & 10 \end{bmatrix}, \quad \tau^B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \]

Using the above algorithm the following sequence of Boolean matrices has been constructed:

\[ q_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad q_2 = \begin{bmatrix} 0 & -\infty \\ -\infty & 0 \end{bmatrix}. \]

The matrix \( q_2 \) satisfies condition \((fS-1)\), so it is the greatest forward simulation between \( \mathcal{A} \) and \( \mathcal{B} \).

If we build the sequence \( \{q_k\}_{k \leq n} \) by means of \((30)\) or \((31)\), the sequence would also stabilize at \( k = 2 \), and we would get the zero matrix (a matrix whose all entries are \(-\infty\)) which satisfies neither \( \sigma^A \leq \sigma^B \cdot q^7 \) nor \( \sigma^B \leq \sigma^A \cdot q \) and \( \tau^A \leq q \cdot \tau^B \). Therefore, there is no forward bisimulation and no backward-forward bisimulation between \( \mathcal{A} \) and \( \mathcal{B} \).

6. Uniform bisimulations

In this section we pay special attention to bisimulations which are uniform Boolean matrices.

Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) and \( \mathcal{B} = (B, \delta^B, \sigma^B, \tau^B) \) be weighted automata. If there exists a row and column complete forward bisimulation between \( \mathcal{A} \) and \( \mathcal{B} \), then \( \mathcal{A} \) and \( \mathcal{B} \) are forward bisimulation equivalent, or briefly \( \mathcal{A} \sim \mathcal{B} \). For arbitrary weighted automata \( \mathcal{A} \), \( \mathcal{B} \) and \( \mathcal{C} \) we have that

\[ \mathcal{A} \sim \mathcal{B} \; \Rightarrow \; \mathcal{A} \sim \mathcal{C} \; \underline{\text{\( \wedge \)}} \; \mathcal{B} \sim \mathcal{C} \; \Rightarrow \; \mathcal{A} \sim \mathcal{C}. \quad (32) \]

Note that row and column completeness of a forward bisimulation mean that every state of \( \mathcal{A} \) is equivalent to some state of \( \mathcal{B} \), and vice versa. As we have shown in Section 4, if there exists at least one forward bisimulation between \( \mathcal{A} \) and \( \mathcal{B} \), then there exists the greatest forward bisimulation which is a partial uniform matrix. In addition, if there is at least one forward bisimulation between \( \mathcal{A} \) and \( \mathcal{B} \) which is row and column complete, then the greatest forward bisimulation is also row and column complete, and hence, it is a uniform Boolean matrix. Therefore, weighted automata \( \mathcal{A} \) and \( \mathcal{B} \) are \( \mathcal{F} \)-equivalent if and only if there is a uniform forward bisimulation between \( \mathcal{A} \) and \( \mathcal{B} \). This fact highlights the importance of studying uniform forward bisimulations, which will be discussed in this section.

The next few results can be proved in a similar way as the corresponding results in [16, 17], so the proofs will be omitted.

**Theorem 6.1.** Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) and \( \mathcal{B} = (B, \delta^B, \sigma^B, \tau^B) \) be weighted automata, and let \( q \in 2^{A \times B} \) be a uniform Boolean matrix. Then \( q \) is a forward bisimulation if and only if the following hold:

(i) \( q^A \) is a forward bisimulation on \( \mathcal{A} \);  
(ii) \( q^B \) is a forward bisimulation on \( \mathcal{B} \);  
(iii) \( q \) is an isomorphism between factor weighted automata \( \mathcal{A} / q^A \) and \( \mathcal{B} / q^B \).

The next theorem describes the case when there exists uniform forward bisimulation between two weighted automata with given kernel and co-kernel.

**Theorem 6.2.** Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) and \( \mathcal{B} = (B, \delta^B, \sigma^B, \tau^B) \) be weighted automata, \( \alpha \) a forward bisimulation on \( \mathcal{A} \) and \( \beta \) a forward bisimulation on \( \mathcal{B} \). Then there exists a uniform forward bisimulation \( q \in 2^{A \times B} \) such that

\[ q^A = \alpha \quad \text{and} \quad q^B = \beta, \quad (33) \]

if and only if the factor weighted automata \( \mathcal{A} / \alpha \) and \( \mathcal{B} / \beta \) are isomorphic.
Proposition 6.3. Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) be a weighted automaton, let \( \alpha, \varrho \in 2^{A \times A} \) be forward bisimulation equivalence matrices, and let \( \eta \in 2^{A \times (A \times \beta)} \) be a Boolean matrix defined by

\[
\eta(a_1, q_{a_2}) = a(a_1, a_2), \quad \text{for all } a_1, a_2 \in A.
\]

Then the following hold.

(A) If \( \varrho < \alpha \), then \( \eta \) is a uniform forward bisimulation between \( \mathcal{A} \) and \( \mathcal{A} / \varrho \) such that \( \eta^A = \alpha \) and \( \eta^{A/\varrho} = \alpha / \varrho \);

(B) If \( \varrho = \alpha \), then \( \eta \) is a uniform forward bisimulation between \( \mathcal{A} \) and \( \mathcal{A} / \alpha \) such that \( \eta^A = \alpha \) and \( \eta^{A/\alpha} \) is the unit matrix.

As a direct consequence of the previous theorem, we obtain that for any forward bisimulation equivalence, the automaton and the corresponding factor automaton are FB-equivalent.

Corollary 6.4. Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) be a weighted automaton, and let \( \alpha \in 2^{A \times A} \) be a forward bisimulation equivalence matrix on \( \mathcal{A} \). Then a Boolean matrix \( \eta \in 2^{A \times (A \times \alpha)} \) defined by

\[
\eta(a_1, a_{a_2}) = a(a_1, a_2), \quad \text{for all } a_1, a_2 \in A,
\]

is a uniform forward bisimulation between \( \mathcal{A} \) and \( \mathcal{A} / \alpha \) such that \( \eta^A = \alpha \) and \( \eta^{A/\alpha} \) is the unit matrix.

The following theorem is also similar to the corresponding theorems from [16, 17], but here we give a different, simpler proof.

Theorem 6.5. Let \( \mathcal{A} = (A, \delta^A, \sigma^A, \tau^A) \) and \( \mathcal{B} = (B, \delta^B, \sigma^B, \tau^B) \) be FB-equivalent weighted automata. Then there exists the greatest forward bisimulation between \( \mathcal{A} \) and \( \mathcal{B} \), which is a uniform Boolean matrix and whose kernel and co-kernel are respectively the greatest forward bisimulation equivalence matrices on \( \mathcal{A} \) and \( \mathcal{B} \).

Proof. By the hypothesis, there exists a uniform forward bisimulation \( \nu \in 2^{A \times B} \), and according to Theorem 4.2, there is the greatest forward bisimulation \( \varrho \in 2^{A \times B} \), which is a partial uniform matrix. Since \( \nu \leq \varrho \) and \( \nu \) is row and column complete, we conclude that \( \varrho \) is also row and column complete, and therefore, \( \varrho \) is a uniform Boolean matrix. According to Theorem 6.1, \( \varrho^A \in 2^{A \times A} \) and \( \varrho^B \in 2^{B \times B} \) are forward bisimulation equivalences on \( \mathcal{A} \) and \( \mathcal{B} \), and \( \varrho \) is an isomorphism between the factor weighted automata \( \mathcal{A} / \varrho^A \) and \( \mathcal{B} / \varrho^B \). Let \( \theta \in 2^{A(\varrho^A)^{-1}(B \times \varrho^B)} \) be a Boolean matrix defined as follows for an arbitrary \( (\gamma_1, \gamma_2) \in (A / \varrho^A) \times (B / \varrho^B) \):

\[
\theta(\gamma_1, \gamma_2) = 1, \quad \text{if } \varrho(\gamma_1) = \gamma_2, \quad \text{and } \theta(\gamma_1, \gamma_2) = 0, \quad \text{otherwise}.
\]

Since \( \varrho \) is an isomorphism, we have that \( \theta \) is a forward bisimulation between \( \mathcal{A} / \varrho^A \) and \( \mathcal{B} / \varrho^B \).

Further, let \( \alpha \in 2^{A \times A} \) and \( \beta \in 2^{B \times B} \) denote the greatest forward bisimulation equivalences on \( \mathcal{A} \) and \( \mathcal{B} \). Let Boolean matrices \( \eta \in 2^{A \times (A \times \varrho^A)} \) and \( \xi \in 2^{B \times (B \times \varrho^B)} \) be defined as in Proposition 6.3, i.e., for arbitrary \( a_1, a_2 \in A \) and \( b_1, b_2 \in B \) we set:

\[
\eta(a_1, a_{a_2}) = a(a_1, a_2), \quad \xi(b_1, b_{a_2}) = \beta(b_1, b_2).
\]

According to Proposition 6.3, \( \eta \) and \( \xi \) are uniform forward bisimulations between \( \mathcal{A} \) and \( \mathcal{A} / \varrho^A \), and \( \mathcal{B} \) and \( \mathcal{B} / \varrho^B \), respectively, and they satisfy

\[
\eta^A = \alpha, \quad \eta^{A / \varrho^A} = \alpha / \varrho^A, \quad \xi^B = \beta, \quad \xi^{B / \varrho^B} = \beta / \varrho^B.
\]

Now, define a Boolean matrix \( \mu \in 2^{A \times B} \) by \( \mu = \eta \cdot \theta \cdot \xi^\top \). Since the composition of forward bisimulations is also a forward bisimulation, we have that \( \mu \) is a forward bisimulation between \( \mathcal{A} \) and \( \mathcal{B} \), and therefore, \( \mu \leq \varrho \). Further, consider arbitrary \( a \in A \) and \( \psi \in F(\varrho) \). Then

\[
\mu(a, \psi(a)) = (\mu \cdot (\varrho^\top \cdot \xi^\top))(a, \psi(a)) = \sum_{a \in A} \eta(a, a_{a_2}) : (\theta \cdot \xi^\top)(\varrho, \psi(a)) = \sum_{a \in A} \eta(a, a_{a_2}) : \xi^\top(\varrho, \psi(a)) = \sum_{a \in A} a(a, a') : \beta(\psi(a), \psi(a')) = \alpha(a, a) \cdot \beta(\psi(a), \psi(a)) = 1,
\]
and therefore,

\[(\mu \cdot \mu^\top)(a, a) = \sum_{b \in B} \mu(a, b) \cdot \mu^\top(b, a) \geq \mu(a, \psi(a)) \cdot \mu^\top(\psi(a), a) = 1,\]

which means that \(\mu \cdot \mu^\top\) is a reflexive Boolean matrix. Seeing that \(\alpha = \eta^A = \eta \cdot \eta^\top\), we obtain that

\[a \cdot \mu = \eta \cdot \eta^\top \cdot \theta \cdot \xi^\top = \eta \cdot \theta \cdot \xi^\top = \mu,\]

and by reflexivity of \(\mu \cdot \mu^\top\) it follows that

\[\alpha \leq a \cdot \mu \cdot \mu^\top = \mu \cdot \mu^\top \leq \eta \cdot \xi^\top = \eta^A.\]

Since both \(\alpha\) and \(\eta^A\) are forward bisimulation equivalences on \(\mathcal{A}\), and \(\alpha\) is the greatest one, we conclude that \(\alpha = \eta^A\), and hence, \(\eta^A\) is the greatest forward bisimulation equivalence on \(\mathcal{A}\).

In the same way we show that \(\beta = \eta^B\), i.e., \(\eta^B\) is the greatest forward bisimulation equivalence on \(\mathcal{B}\).

As an immediate consequence of Theorems 6.5, 6.1 and 6.2 we obtain the following:

**Corollary 6.6.** Let \(\mathcal{A} = (A, \delta^A, \sigma^A, \tau^A)\) and \(\mathcal{B} = (B, \delta^B, \sigma^B, \tau^B)\) be weighted automata, and let \(\alpha\) and \(\beta\) be respectively the greatest forward bisimulation equivalence matrices on \(\mathcal{A}\) and \(\mathcal{B}\). Then \(\mathcal{A}\) and \(\mathcal{B}\) are FB-equivalent if and only if factor automata \(\mathcal{A}/\alpha\) and \(\mathcal{B}/\beta\) are isomorphic.

Now we prove the following result.

**Theorem 6.7.** Let \(\mathcal{A}\) be a weighted automaton, let \(\alpha\) be the greatest forward bisimulation equivalence matrix on \(\mathcal{A}\), and let \(\mathcal{FB}(\mathcal{A})\) be the class of all weighted automata which are FB-equivalent to \(\mathcal{A}\). Then the factor weighted automaton \(\mathcal{A}/\alpha\) is the unique (up to an isomorphism) minimal automaton in \(\mathcal{FB}(\mathcal{A})\).

**Proof.** According to Corollary 6.4, \(\mathcal{A}/\alpha\) belongs to \(\mathcal{FB}(\mathcal{A})\). Let \(\mathcal{B}\) be any minimal weighted automaton from \(\mathcal{FB}(\mathcal{A})\), and let \(\beta\) be the greatest forward bisimulation equivalence matrix on \(\mathcal{B}\). By Corollary 6.6 it follows that \(\mathcal{B}/\beta \cong \mathcal{A}/\alpha\), so \(\mathcal{B}/\beta\) also belongs to \(\mathcal{FB}(\mathcal{A})\), and by minimality of \(\mathcal{B}\) we obtain that \(\beta\) is the unit matrix. Therefore, \(\mathcal{B} \cong \mathcal{B}/\beta \cong \mathcal{A}/\alpha\), proving our claim. \(\square\)

Results analogous to those which have been proved here for forward bisimulations can also be proved for backward bisimulations. However, in the case of backward-forward and forward-backward bisimulations the situation is somewhat different. The main difference between homotypic and heterotypic bisimulations is as follows. If \(\varphi\) is a forward or backward bisimulation, then \(\varphi^\top\) has the same property, but if \(\varphi\) is a backward-forward bisimulation, then \(\varphi^\top\) is a forward-backward bisimulation, and vice versa.

Moreover, the following is true.

**Theorem 6.8.** Let \(\mathcal{A} = (A, \delta^A, \sigma^A, \tau^A)\) and \(\mathcal{B} = (B, \delta^B, \sigma^B, \tau^B)\) be weighted automata, and let \(\varphi \in 2^{A \times B}\) be a uniform Boolean matrix. Then \(\varphi\) is backward-forward bisimulation if and only if the following is true:

(i) \(\varphi^A\) is a forward bisimulation on \(\mathcal{A}\);
(ii) \(\varphi^B\) is a backward bisimulation on \(\mathcal{B}\);
(iii) \(\varphi\) is an isomorphism between factor weighted automata \(\mathcal{A}/\varphi^A\) and \(\mathcal{B}/\varphi^B\).

Finally, we prove that when working with functional matrices, the concepts of a forward bisimulation and a backward-forward bisimulation coincide.

**Theorem 6.9.** Let \(\mathcal{A} = (A, \delta^A, \sigma^A, \tau^A)\) and \(\mathcal{B} = (B, \delta^B, \sigma^B, \tau^B)\) be weighted automata, and let \(\varphi \in 2^{A \times B}\) be a functional matrix. Then the following conditions are equivalent:

(i) \(\varphi\) is a forward bisimulation;
(ii) \(\varphi\) is a backward-forward bisimulation.
(iii) \( \varphi^A \) is a forward bisimulation on \( \mathcal{A} \) and the function \( \phi: A/\varphi^A \to B \) given by \( \phi(\varphi^A_a) = \varphi(a) \), for each \( a \in A \), is a monomorphism of the factor weighted automaton \( \mathcal{A}/\varphi^A \) into \( \mathcal{B} \).

Proof. Let \( C = \text{Im} \varphi = \{ b \in B | (\exists a \in A) \varphi(a, b) = 1 \} \), and consider the subautomaton \( \mathcal{C} = (C, \delta^C, \sigma^C, \tau^C) \) of \( \mathcal{B} \). Clearly, we can assume that \( \varphi \in 2^{A \times C} \).

(i)\( \Rightarrow \) (iii). Let (i) hold. Then \( \varphi \) is a forward bisimulation from \( \mathcal{A} \) to \( \mathcal{C} \). We also have that \( \varphi \) is a column complete Boolean matrix (surjective function), and hence, it is a uniform matrix. Now, by Theorem 6.1 we obtain that \( \varphi^A \) is a forward bisimulation on \( \mathcal{A} \). \(\varphi^C \) is the unit matrix, and \( \varphi \) is an isomorphism from \( \mathcal{A}/\varphi^A \) to \( \mathcal{C}/\varphi^C \cong \mathcal{C} \). If we identify \( \mathcal{C}/\varphi^C \) and \( \mathcal{C} \), then it is easy to see that \( \varphi \) can be represented as \( \phi \), where \( \phi \) is defined as in (iii), so \( \varphi \) is a monomorphism of \( \mathcal{A}/\varphi^A \) into \( \mathcal{B} \).

(iii)\( \Rightarrow \) (i). This is a direct consequence of Theorem 6.1, since \( \varphi^C \) is the unit matrix and \( \varphi \) and \( \phi \) can be identified.

(i)\( \Leftrightarrow \) (ii). This follows immediately by Theorems 6.1 and 6.8, since \( \varphi^C \) is the unit matrix, and it is both a forward and backward bisimulation equivalence.

References