Journal of Automata, Languages and Combinatorics **8** (2003) 2, 187–218

WEAK BISIMULATION FOR (max /+) AUTOMATA AND RELATED MODELS [1]

Peter Buchholz[2]
Fakultät für Informatik, TU Dresden
D-01062 Dresden, Germany
e-mail: p.buchholz@inf.tu-dresden.de

and

Peter Kemper[3]
Informatik IV, Universität Dortmund
D-44221 Dortmund, Germany
e-mail: peter.kemper@udo.edu

ABSTRACT

In this paper we propose a notion of weak bisimulation for a class of weighted automata with unobservable transitions where weights are taken from an idempotent semiring. In particular the (max/+), (min/+) or (max/min) dioid are considered. The proposed bisimulation is a natural extension of Milner's well known weak bisimulation for untimed automata (i. e., weighted automata over the boolean semiring). It is shown that weakly equivalent automata yield identical results with respect to the weights of arbitrarily labelled paths. The basic steps of an algorithm for the computation of the largest weak bisimulation relation for a given weighted automaton is outlined. Furthermore we present composition operations for weighted automata and prove that weak bisimulation is a congruence according to the composition operations presented in this paper.

Keywords: Weighted automata, dioids, weak bisimulation, composition

1. Introduction

One topic of automata theory considers conditions, criteria and circumstances under which two automata can be considered equivalent. For instance, observing the behaviour of an automata and to compare the traces or languages produced is one possibility. In case of an interacting environment, more appropriate notions of equivalence than trace-equivalence are necessary and established. Milner's classical results consider strong and weak bisimulation in the context of the process algebra CCS

[1] Full version of a lecture presented at the Workshop *Weighted Automata: Theory and Applications* (Dresden University of Technology, Germany, March 4 – 8, 2002).

[2] This research is partially supported by DFG, SFB 358.

[3] This research is partially supported by DFG, SFB 559.

[21, 25]. If systems under study interact with an observer or other systems, the notion of secretly performed, unobservable, so-called τ actions appears naturally for those actions a system is able to perform independently of any environment. Apart from the aspect of information hiding, τ actions also introduce a degree of freedom to find minimal representations of an automaton. This is of even greater interest in case of subsequent composition in order to reduce the complexity of analysis for the resulting overall model. A notion of equivalence that is stable under a set of composition operators is called congruence, and congruences like Milner's observation congruence have been developed and successfully applied in functional analysis [10].

Equivalence results for untimed models have been subsequently extended to more general models, especially to probabilistic [20] and stochastic systems [5, 17, 16]. In the context of probabilistic and stochastic process algebras it has been shown that strong bisimulation as a natural extension of Milner's strong bisimulation is a congruence and preserves the probabilistic and timed behaviour of a system. The extension of weak bisimulation to more general models is much more complex, than the relatively straightforward extension of strong bisimulation. The major problem is that unobservable transitions in probabilistic or stochastic systems still may have effects on the observable behaviour of a system by modifying the timed or probabilistic behaviour of observable transitions. To the best of our knowledge no notion of weak bisimulation exists for stochastic systems, if unobservable transitions may require time like observable transitions. For probabilistic systems a few extensions of weak and branching bisimulation exist [4, 1]. However, the problem of these extensions is that they are not preserved under parallel composition which is an important operation in process algebras and networks of automata.

In the context of weighted automata [11, 19] probabilistic and stochastic automata can be interpreted as an extension of untimed automata which results from using another semiring to define transition weights. Recently it has been shown that strong bisimulation can be easily extended to weighted automata and is preserved by the common operations to compose automata [8]. To the best of our knowledge no general extension of weak bisimulation has been proposed for this general class of models. In this paper we go one small step in defining weak bisimulation for a class of weighted automata. We present a definition of weak bisimulation for a class of weighted automata defined over idempotent and commutative semirings. The boolean semiring is an example for such a semiring and, consequently, our definition extends weak bisimulation to more general models and is identical to the well known definition of weak bisimulation when applied to automata with weights taken from the boolean semiring. We show that for this class of weighted automata weak bisimulation is preserved under various composition operations and preserves the relevant results of system analysis.

The class of weighted automata over idempotent and commutative semirings includes, apart from the boolean case, additionally several very interesting semirings like $\max/+$, $\min/+$ or $\min/\max$ systems resulting in different automata models which have been used for a variety of analysis problems related to performance analysis of discrete event systems, combinatorial optimization or language processing [3, 13, 22]. Typical idempotent semirings are used $\mathbb{R}_{\max} = (\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)$, $\mathbb{R}_{\min} =$

$(\mathbb{R} \cup \{\infty\}, \min, +, \infty, 0)$, $\mathbb{R}_{\max,\min} = (\mathbb{R} \cup \{\pm\infty\}, \max, \min, -\infty, \infty)$ and their completed versions $\overline{\mathbb{R}}_{\max} = (\mathbb{R} \cup \{\pm\infty\}, \max, +, -\infty, 0)$, $\overline{\mathbb{R}}_{\min} = (\mathbb{R} \cup \{\pm\infty\}, \min, +, \infty, 0)$.

The class of automata we consider does not include probabilistic or stochastic automata which are defined over a non-idempotent semiring. Idempotency of the semiring is sufficient to assure preservation of weak bisimulation under parallel composition. Nevertheless, our definition of weak bisimulation can in principle be applied to automata over general semiring and yields for probabilistic automata a definition which is identical to the one presented in [4] for probabilistic processes.

The paper is structured as follows. In Section 2, we fix some basic definitions. Section 3 presents the main result of the paper, the weak bisimulation for a class of weighted automata. The congruence property of the proposed equivalence is proven in Section 4. In Section 5 we consider weak bisimulation under a choice operator. Section 6 illustrates the approach by several small application examples and we conclude in Section 7.

2. A Class of Weighted Automata

Weighted automata are a well established extension of classical automata [19] which are applied in various variants for different application areas like performance analysis of discrete event systems [13], language processing [22] or image compression [18] to mention only a few application areas. Transition weights in weighted automata are defined over a semiring.

Definition 2.1 (Semiring) *A semiring* $(\mathbb{K}, \hat{+}, \hat{\cdot}, \mathbb{0}, \mathbb{1})$ *is a set* $\mathbb{K}$ *with binary operations* $\hat{+}$ *and* $\hat{\cdot}$ *defined on* $\mathbb{K}$ *such that the following axioms are satisfied:*

1. $\hat{+}$, $\hat{\cdot}$ *are associative,*
2. $\hat{+}$ *is commutative,*
3. *right and left distributive laws hold for* $\hat{+}$ *and* $\hat{\cdot}$*,*
4. $\mathbb{0}$ *and* $\mathbb{1}$ *are the additive and multiplicative identities with* $\mathbb{0} \neq \mathbb{1}$*,*
5. *for all* $k \in \mathbb{K}$ $k \hat{\cdot} \mathbb{0} = \mathbb{0} \hat{\cdot} k = \mathbb{0}$ *holds.*

To make the notation simpler we use sometimes $\mathbb{K}$ for the whole semiring. For notational convenience often ab is used for $a \hat{\cdot} b$. If multiplication is also commutative, then the semiring is commutative and if addition is idempotent, then the semiring is idempotent. The semirings of the $\max/+$ family mentioned in the introduction are commutative and idempotent whereas the semiring for the definition of probabilistic automata, using the normal addition and multiplication operations, is commutative but not idempotent.

Matrices and vectors can be naturally defined using the elements and operations of a semiring $\mathbb{K}$. Let $\mathbf{A}, \mathbf{B} \in \mathbb{K}^{n,n}$ be two matrices, then

$$\mathbf{C} = \mathbf{A} \hat{+} \mathbf{B} \quad \text{with } \mathbf{C} \in \mathbb{K}^{n,n} \text{ and } \mathbf{C}(i,j) = \mathbf{A}(i,j) \hat{+} \mathbf{B}(i,j),$$
$$\mathbf{D} = \mathbf{A} \hat{\cdot} \mathbf{B} \quad \text{with } \mathbf{D} \in \mathbb{K}^{n,n} \text{ and } \mathbf{D}(i,j) = \hat{\sum}_{k=1}^{n} \mathbf{A}(i,k) \hat{\cdot} \mathbf{B}(k,j),$$

where $\widehat{\sum}_{k=1}^{n} a_k = a_1 \mathbin{\widehat{+}} \cdots \mathbin{\widehat{+}} a_n$. For $\mathbf{A} \in \mathbb{K}^{n,n}$ and $\mathbf{B} \in \mathbb{K}^{m,m}$ the Kronecker product and sum are defined as

$$\mathbf{C} = \mathbf{A} \mathbin{\widehat{\otimes}} \mathbf{B} \in \mathbb{K}^{nm,nm} \quad \text{with } \mathbf{C} = \begin{pmatrix} \mathbf{A}(1,1) \mathbin{\widehat{\cdot}} \mathbf{B} & \dots & \mathbf{A}(1,n) \mathbin{\widehat{\cdot}} \mathbf{B} \\ \vdots & \ddots & \vdots \\ \mathbf{A}(n,1) \mathbin{\widehat{\cdot}} \mathbf{B} & \dots & \mathbf{A}(n,n) \mathbin{\widehat{\cdot}} \mathbf{B} \end{pmatrix},$$

$$\mathbf{D} = \mathbf{A} \mathbin{\widehat{\oplus}} \mathbf{B} \in \mathbb{K}^{n,n} \qquad \text{with } \mathbf{D} = \mathbf{A} \mathbin{\widehat{\otimes}} \mathbf{I}_m \mathbin{\widehat{+}} \mathbf{I}_n \mathbin{\widehat{\otimes}} \mathbf{B}.$$

where $\mathbf{I}_n$ is the identity matrix of order n over semiring $\mathbb{K}$ which is a matrix with $\mathbb{1}$ at the main diagonal and $\mathbb{0}$ elsewhere.

Lemma 2.1 *If multiplication is commutative, then*

$$\left(\mathbf{A} \mathbin{\widehat{\otimes}} \mathbf{B}\right) \mathbin{\widehat{\cdot}} \left(\mathbf{A} \mathbin{\widehat{\otimes}} \mathbf{B}\right) = \left(\mathbf{A} \mathbin{\widehat{\cdot}} \mathbf{A}\right) \mathbin{\widehat{\otimes}} \left(\mathbf{B} \mathbin{\widehat{\cdot}} \mathbf{B}\right).$$

For commutative and idempotent semirings we have additionally

$$\left(\mathbf{A} \mathbin{\widehat{\otimes}} \mathbf{I} \mathbin{\widehat{+}} \mathbf{I} \mathbin{\widehat{\otimes}} \mathbf{B}\right)^2 = \mathbf{A}^2 \mathbin{\widehat{\otimes}} \mathbf{I} \mathbin{\widehat{+}} \mathbf{A} \mathbin{\widehat{\otimes}} \mathbf{B} \mathbin{\widehat{+}} \mathbf{I} \mathbin{\widehat{\otimes}} \mathbf{B}^2.$$

By induction this can be extended to the relation

$$\left(\mathbf{A} \mathbin{\widehat{\otimes}} \mathbf{I} \mathbin{\widehat{+}} \mathbf{I} \mathbin{\widehat{\otimes}} \mathbf{B}\right)^k = \widehat{\sum_{l=0}^{k}} \mathbf{A}^l \mathbin{\widehat{\otimes}} \mathbf{B}^{k-l}.$$

For idempotent semirings also

$$\widehat{\sum_{k=0}^{\infty}} \mathbf{A}^k = \widehat{\sum_{k=0}^{\infty}} \mathbf{A}^k \mathbin{\widehat{\cdot}} \widehat{\sum_{k=0}^{\infty}} \mathbf{A}^k$$

holds.

Proof. The first three statements can be proven by definition of the algebraic operations and induction, the proof is straightforward. For the final statement, we can transform $\widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k \mathbin{\widehat{\cdot}} \widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k = \widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k \mathbin{\widehat{+}} \widehat{\sum}_{k=1}^{\infty} \mathbf{A}^k \mathbin{\widehat{\cdot}} \widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k = \widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k \mathbin{\widehat{+}} \widehat{\sum}_{k=1}^{\infty} \mathbf{A}^k \mathbin{\widehat{+}} \cdots \mathbin{\widehat{+}} \widehat{\sum}_{k=n}^{\infty} \mathbf{A}^k \mathbin{\widehat{\cdot}} \widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k$ for some $n \in \mathbb{N}$, $n > 1$ by distributivity. Note that idempotency ensures that $\widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k \mathbin{\widehat{+}} \mathbf{A}^n = \widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k$ for all $n \in \mathbb{N}$, such that also $\widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k \mathbin{\widehat{+}} \widehat{\sum}_{k=n}^{\infty} \mathbf{A}^k = \widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k$ for all $n \in \mathbb{N}$. Hence by applying idempotency to $\widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k \mathbin{\widehat{+}} \widehat{\sum}_{k=1}^{\infty} \mathbf{A}^k \mathbin{\widehat{+}} \cdots \mathbin{\widehat{+}} \widehat{\sum}_{k=n}^{\infty} \mathbf{A}^k \mathbin{\widehat{\cdot}} \widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k = \widehat{\sum}_{k=0}^{\infty} \mathbf{A}^k$. □

Observe that the conditions of the lemma are necessary conditions. I. e., for non-commutative semirings the first identity does not hold, the fourth identity does not hold for semirings that are not idempotent and for semirings that are not commutative or not idempotent, the second and third identity do not hold.

Definition 2.2 ($\mathbb{K}$-automaton) *A finite $\mathbb{K}$-automaton over a finite alphabet $\mathcal{L}$ including symbol τ is a quadruple $\mathcal{A} = (\mathcal{S}, \alpha, T, \beta)$ where $\mathcal{S}$ is a finite set of states and*

α, T, β are maps $\alpha : \mathcal{S} \to \mathbb{K}$, $T : \mathcal{S} \times \mathcal{L} \times \mathcal{S} \to \mathbb{K}$, $\beta : \mathcal{S} \to \mathbb{K}$, denoted as initial-, transition- and final-weights, respectively. If $\mathbb{K}$ is idempotent and commutative, we denote the automaton as a $\mathbb{K}i$-automaton.

In the following, we restrict ourselves to $\mathbb{K}i$-automata. For finite sets $\mathcal{S}$ and $\mathcal{L}$, map T has a representation by a set of matrices $\mathbf{M}_l \in \mathbb{K}^{\mathcal{S}\times\mathcal{S}}$, one for each $l \in \mathcal{L}$ with $\mathbf{M}_l(s, s') = T(s, l, s')$. Maps α, β have an equivalent representation as row and column vectors[4] $\mathbf{a}, \mathbf{b}^T \in \mathbb{K}^{\mathcal{S}}$ with $\mathbf{a}(s) = \alpha(s), \mathbf{b}(s) = \beta(s)$ for all $s \in \mathcal{S}$.

Definition 2.3 (Paths in $\mathbb{K}$-automata) *A path π of length n in a $\mathbb{K}$-automaton is a sequence $s_0, l_1, s_1, \ldots, l_n, s_n$ with $s_i \in \mathcal{S}$, $l_j \in \mathcal{L}$, and $T(s_{i-1}, l_i, s_i) \neq \mathbb{0}$. Let $\sigma = l_1, \ldots, l_n$ denote a sequence of labels in $\mathcal{L}^*$. $\mathcal{P}(\mathcal{A})$ is the set of all paths of automaton $\mathcal{A}$ and $\mathcal{P}(\mathcal{A}, \sigma)$ is the set of all paths observing transition labels σ. The weight of a path π is defined as*

$$w(\pi) = \alpha(s_0) \mathbin{\widehat{\cdot}} \left(\widehat{\prod_{i=1}^{n}} T(s_{i-1}, l_i, s_i) \right) \mathbin{\widehat{\cdot}} \beta(s_n) \tag{1}$$

and the weight of sequence σ equals

$$w(\sigma) = \widehat{\sum_{\pi \in \mathcal{P}(\mathcal{A},\sigma)}} w(\pi) = \mathbf{a} \mathbin{\widehat{\cdot}} \widehat{\prod_{i=1}^{n}} \mathbf{M}_{l_i} \mathbin{\widehat{\cdot}} \mathbf{b}. \tag{2}$$

As usual empty products are $\mathbb{1}$ and empty sums are $\mathbb{0}$. Transitions labelled with τ are not visible by an observer. Following the idea proposed in [2, 21], we can transform our original transition system into a new one where τ-labelled transitions are skipped. Let $\mathcal{P}(\mathcal{A}, s, \sigma, s')$ be the set of all paths which start in state s, end in state s' and with a sequence of labels σ. Let τ^* denote a set of sequences that consist of τ transitions only, including the empty sequence as usual. We define

$$T(s, \tau^*, s') = \begin{cases} \mathbb{1} \mathbin{\widehat{+}} \widehat{\sum_{n=1}^{\infty}} \widehat{\sum_{\pi \in \mathcal{P}(\mathcal{A},s,(\tau_1,\ldots,\tau_n),s')}} \left(\widehat{\prod_{i=1}^{n}} T(s_{i-1}, \tau, s_i) \right) & \text{if } s = s', \\ \widehat{\sum_{n=1}^{\infty}} \widehat{\sum_{\pi \in \mathcal{P}(\mathcal{A},s,(\tau_1,\ldots,\tau_n),s')}} \left(\widehat{\prod_{i=1}^{n}} T(s_{i-1}, \tau, s_i) \right) & \text{otherwise,} \end{cases} \tag{3}$$

and $T(s, \tau^* l \tau^*, s') = \widehat{\sum}_{s_1,s_2\in\mathcal{S}} T(s, \tau^*, s_1) T(s_1, l, s_2) T(s_2, \tau^*, s')$. For notational convenience, we extend some definitions to sets. For a set of states $C \subseteq \mathcal{S}$, let $T(s, l, C) = \widehat{\sum}_{c\in C} T(s, l, c)$, $\mathbf{M}_l(s, C) = \widehat{\sum}_{c\in C} \mathbf{M}_l(s, c)$, $\alpha(C) = \widehat{\sum}_{c\in C} \alpha(c)$, and $\beta(C) = \widehat{\sum}_{c\in C} \beta(c)$.

Definition 2.4 (Automaton $\mathcal{A}^*$) *For a given $\mathbb{K}$-automaton $\mathcal{A} = (\mathcal{S}, \alpha, T, \beta)$, automaton $\mathcal{A}^* = (\mathcal{S}, \alpha, T', \beta')$ is defined over alphabet $\mathcal{L}' = \mathcal{L}\backslash\{\tau\} \cup \{\epsilon\}$ with $T'(s, l, s') = T(s, \tau^* l \tau^*, s')$ and $\beta'(s) = \widehat{\sum}_{s'\in\mathcal{S}} T(s, \tau^*, s')\beta(s')$. For ϵ, $T'(s, \epsilon, s') = T(s, \tau^*, s')$. Furthermore, $T'(s, \epsilon, s) = \mathbb{1}$ for a state s if s has no outgoing τ-transitions in $\mathcal{A}$.*

[4] $\mathbb{K}^{\mathcal{S}}$ is the set of row vectors over $\mathcal{S}$ with elements in $\mathbb{K}$.

Observe that ϵ in $\mathcal{A}^*$ substitutes τ^* in $\mathcal{A}$, which abbreviates sequences of τ transitions but still allows one to identify unobservable transitions in $\mathcal{A}^*$. The existence of $T(s, \tau^*, s')$ depends on the infinite sum in (3) which is crucial for the existence of $\mathcal{A}^*$. If there are no cyclic paths of τ-labelled transitions, it is achieved by a finite summation. If such cycles exist, the semiring needs to be closed to ensure that infinite addition is defined and behaves like finite addition, e. g., cycles of τ-labelled transitions with positive weights yield weight ∞ in $T(s, \tau^*, s')$ for semiring $\overline{\mathbb{R}}_{\max}$ and can not be computed in $\mathbb{R}_{\max}$.

The notion of $T(s, \tau^*, s')$ can be equivalently described for all $s, s' \in \mathcal{S}$ by a matrix $\mathbf{M}'_\epsilon = \mathbf{M}^*_\tau = \widehat{\sum}_{i=0}^{\infty} \mathbf{M}^i_\tau$. Note that in matrix terms, the definition of T' yields $\mathbf{M}'_l = \mathbf{M}^*_\tau \mathbin{\widehat{\cdot}} \mathbf{M}_l \mathbin{\widehat{\cdot}} \mathbf{M}^*_\tau$ and the definition of β' yields $\mathbf{M}^*_\tau \mathbf{b}$.

Similar to [21], we define for $\sigma \in \mathcal{L}^*$, $\hat{\sigma} \in \mathcal{L}'^*$ as the sequence which results from σ by deleting all occurrences of τ.

Definition 2.5 *For a $\mathbb{K}$-automaton $\mathcal{A}^*$ and $\hat{\sigma} \in \mathcal{L}'^*$, the weight of $\hat{\sigma}$ is defined as*

$$w^*(\hat{\sigma}) = \widehat{\sum_{\varsigma \in \mathcal{L}^* : \hat{\varsigma} = \hat{\sigma}}} w(\varsigma) = \mathbf{a} \mathbin{\widehat{\cdot}} \widehat{\prod_{i=1}^{n}} (\mathbf{M}^*_\tau \mathbin{\widehat{\cdot}} \mathbf{M}_{l_i}) \mathbin{\widehat{\cdot}} \mathbf{M}^*_\tau \mathbin{\widehat{\cdot}} \mathbf{b}.$$

For idempotent semiring path weights can be computed by composing the corresponding values of T', e. g., for l, l' the concatenation yields $\mathbf{M}^*_\tau \mathbin{\widehat{\cdot}} \mathbf{M}_l \mathbin{\widehat{\cdot}} \mathbf{M}^*_\tau \mathbin{\widehat{\cdot}} \mathbf{M}^*_\tau \mathbin{\widehat{\cdot}} \mathbf{M}_{l'} \mathbin{\widehat{\cdot}} \mathbf{M}^*_\tau = \mathbf{M}^*_\tau \mathbin{\widehat{\cdot}} \mathbf{M}_l \mathbin{\widehat{\cdot}} \mathbf{M}^*_\tau \mathbin{\widehat{\cdot}} \mathbf{M}_{l'} \mathbin{\widehat{\cdot}} \mathbf{M}^*_\tau$. If the semiring is not idempotent, then paths weights in general can not be computed by concatenation of T' because $\mathbf{M}^*_\tau \mathbf{M}^*_\tau \neq \mathbf{M}^*_\tau$.

Lemma 2.2 *If $\mathcal{A}$ contains no cycles of τ-labelled transitions, then $\mathcal{A}^*$ can be computed with an effort in $O(n^3)$ where $n = |\mathcal{S}|$ the number of states.* □

The lemma follows from standard results in $\max/+$-algebra (e. g., [3, 14]). For automata with non-negative weights, a computation of shortest paths between states in case of $\overline{\mathbb{R}}_{\min}$ and a computation of $\mathcal{A}^*$ for $\overline{\mathbb{R}}_{\max}$ is possible in $O(n^3)$ as well. For the latter, cycles in the graph are considered and yield weight ∞. In the general case, treatment of cycles makes computation of $\mathcal{A}^*$ more complex. Algorithms for the removal of τ-transitions can be found in the software library for weighted transducers [23] where τ is denoted as ϵ. These algorithms can be extended to be used for the computation of T'. For several semirings like the tropical semiring (i. e., $\mathbb{K} = (\mathbb{N} \cup \{+\infty\}, \min, +, \infty, 0)$) shortest path algorithms are the most efficient approaches to compute $\mathbf{M}^*_\tau$, for other semirings the well known Floyd-Warshall algorithm can be applied as a generic algorithm for the computation of the transitive closure of τ-transitions. Consequently, in several cases, computation of $\mathbf{M}^*_\tau$ requires less than $O(n^3)$ effort.

3. Weak Bisimulation for Weighted Automata

After the definition of weighted automata and the introduction of a transformation to skip τ-transitions, we now consider equivalence of automata. In [6, 8], the notion

of strong bisimulation has been extended to weighted automata. This equivalence, like strong bisimulation for automata over the boolean semiring [21], assures that every transition labelled with a in one automaton has to be matched by one or more a-labelled transitions with the same weight in an equivalent automaton. This condition is required even for τ-labelled transitions. Here, we relax the requirement by not observing τ-labelled transitions. However, unobservable transitions may still induce cost which is different from the boolean semiring where only the existence or non-existence of a connection by unobservable transitions has to be considered. In this section, we extend the definition of weak bisimulation to our class of weighted automata and prove several properties of weak bisimulation. The following definition introduces weak bisimulation for the states of one automaton as well as weak bisimulation for two automata, but requires some notation prerequisites.

For the definition of weakly equivalent automata we need the direct sum of two automata which is defined formally in Definition 4.1. The direct sum $\mathcal{A}_1 + \mathcal{A}_2$ generates a new automaton $\mathcal{A}$ with $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$. We implicitly assume throughout the paper that $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$. Let $\mathcal{S}/\mathcal{R}$ denote the set of equivalence classes for an equivalence relation $\mathcal{R}$ over $\mathcal{S}$ and for any class $C \in \mathcal{S}/\mathcal{R}$, let $C_i = C \cap \mathcal{S}_i$ for some set $\mathcal{S}_i \subseteq \mathcal{S}$.

Definition 3.1 (Weak bisimulation of $\mathbb{K}$-automata) *An equivalence relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$ is a weak bisimulation relation, iff for all $(s_1, s_2) \in \mathcal{R}$, all $l \in \mathcal{L}'$ and all equivalence classes $C \in \mathcal{S}/\mathcal{R}$*

$$\alpha(s_1) = \alpha(s_2),\ \beta'(s_1) = \beta'(s_2),\ \text{and } T'(s_1, l, C) = T'(s_2, l, C).$$

Two states s_1 and s_2 are weakly bisimilar denoted as $s_1 \approx s_2$, if a weak bisimulation relation $\mathcal{R}$ with $(s_1, s_2) \in \mathcal{R}$ exists.

Two automata $\mathcal{A}_1$ and $\mathcal{A}_2$ are weakly bisimilar denoted as $\mathcal{A}_1 \approx \mathcal{A}_2$, if for $\mathcal{A} = \mathcal{A}_1 + \mathcal{A}_2$ with state space $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$, a weak bisimulation relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$ exists such that

$$\alpha(C_1) = \alpha(C_2) \text{ for all } C \in \mathcal{S}/\mathcal{R}. \qquad (*)$$

The idea behind weak bisimulation is that in weakly bisimilar automata those paths that have the same sequence of observable labels $l_i \neq \tau$ shall have the same weights. The conditions might seem too restrictive due to the required identity of α and β' which is not explicitly available in the definition of weak bisimulation for CCS. However, in CCS and related process algebras states are unobservable and paths are only observed through transitions. This viewpoint can be adopted in our automata model, which yields $\beta(s) = \mathbb{1}$ for all states. Thus, $\beta'(s) = \beta'(s')$ is observed if $T'(s, \epsilon, C) = T'(s', \epsilon, C)$ for all equivalence classes C of some weak bisimulation $\mathcal{R}$. The condition $(*)$ on the identity of α values for two automata allows us to consider automata with several initial states with possibly different weights; it is used in the proof of Theorem 3.1 below. The additional condition of identical values $\alpha(s)$ for all states in an equivalence class is for instance required if the automaton is used as second automaton in a direct product as defined in the next section or if initial weights are required for the computation of path weights. If initial weights are not used in a model, their values can be appropriately set to make them irrelevant, e. g., $\alpha(s)$ can

be set to $\mathbb{1}$ for all states such that the condition of identical values is observed for all states; in case of an idempotent semiring, this then also fulfills $(*)$ if sets C_1, C_2 are non-empty. The conditions for weak bisimulation given in Definition 3.1 can be equivalently expressed using the matrices and vectors that define $\mathcal{A}$. Let $\mathcal{R}$ be some weak bisimulation, then we have for all $(s, s') \in \mathcal{R}$ and for all equivalence classes $C \in \mathcal{S}/\mathcal{R}$

$$\mathbf{a}(s) = \mathbf{a}(s'),\ \mathbf{a} \mathbin{\widehat{\cdot}} (\mathbf{e}_{C_1})^T = \mathbf{a} \mathbin{\widehat{\cdot}} (\mathbf{e}_{C_2})^T, \tag{4}$$

$$\mathbf{e}_s \mathbf{M}_\tau^* \mathbf{b} = \mathbf{e}_{s'} \mathbf{M}_\tau^* \mathbf{b}, \tag{5}$$

$$\mathbf{e}_s \mathbf{M}_\tau^* \mathbf{M}_l \mathbf{M}_\tau^* (\mathbf{e}_C)^T = \mathbf{e}_s \mathbf{M}_l^{'} (\mathbf{e}_C)^T = \mathbf{e}_{s'} \mathbf{M}_l^{'} (\mathbf{e}_C)^T = \mathbf{e}_{s'} \mathbf{M}_\tau^* \mathbf{M}_l \mathbf{M}_\tau^* (\mathbf{e}_C)^T, \tag{6}$$

$$\mathbf{e}_s \mathbf{M}_\tau^* (\mathbf{e}_C)^T = \mathbf{e}_s \mathbf{M}_\epsilon^{'} (\mathbf{e}_C)^T = \mathbf{e}_{s'} \mathbf{M}_\epsilon^{'} (\mathbf{e}_C)^T = \mathbf{e}_{s'} \mathbf{M}_\tau^* (\mathbf{e}_C)^T, \tag{7}$$

where $\mathbf{e}_s$ is a row vector of length $|\mathcal{S}|$ with $\mathbb{1}$ in position s and $\mathbb{0}$ elsewhere and $\mathbf{e}_C$ is a row vector of length $|\mathcal{S}|$ with $\mathbb{1}$ in all positions $s \in C$ and $\mathbb{0}$ elsewhere.

The following theorem states that weights of observable sequences are equal for weakly bisimilar $\mathbb{K}i$-automata.

Theorem 3.1 *If $\mathcal{A}_1 \approx \mathcal{A}_2$ for $\mathbb{K}i$-automata $\mathcal{A}_1, \mathcal{A}_2$, then $w_1^*(\hat{\sigma}) = w_2^*(\hat{\sigma})$ for all $\hat{\sigma} \in \mathcal{L}'^*$ ($\mathcal{L}' = \mathcal{L}_1' \cup \mathcal{L}_2'$), where $w_i^*(\hat{\sigma})$ is the weight of sequence $\hat{\sigma} = l_1, \ldots, l_n$ in automaton $\mathcal{A}_i$.*

Proof. The proof is done by induction on n, the length of sequences $\hat{\sigma}$ with respect to labels in $\mathcal{L}'$. So, n accounts only for labels $\neq \tau$.

For a sequence with $n = 0$, i. e., an empty sequence or a sequence of τ transitions, the result is immediate by definition of β'. For a sequence with $n = 1$, the hypothesis is ensured by the definition of weak bisimulation.

We consider a sequence with $n > 1$. By definition of weak bisimulation, all elements s, s' of an equivalence class $C \in \mathcal{S}/\mathcal{R}$ have the same values $\alpha(s) = \alpha(s')$ and $\beta'(s) = \beta'(s')$. We define a sequence of vectors $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n$ by $\mathbf{v}_n = \mathbf{b}'$ and $\mathbf{v}_i = \mathbf{M}_l' \mathbin{\widehat{\cdot}} \mathbf{v}_{i+1}$ for $i = 0, 1, 2, \ldots, n-1$. By definition $\mathbf{v}_n(s) = \mathbf{v}_n(s')$ for all s, s' of an equivalence class C. Property $T'(s_1, l, C) = T'(s_2, l, C)$ in Def. 3.1 preserves this condition for vector $\mathbf{v}_{n-1} = \mathbf{M}_{l_n}' \mathbin{\widehat{\cdot}} \mathbf{b}'$. To check this in detail, we consider states $(s_1, s_2) \in \mathcal{R}$ and values $\mathbf{v}_{n-1}(s_1), \mathbf{v}_{n-1}(s_2)$.

$$\begin{aligned}
\mathbf{v}_{n-1}(s_1) &= \widehat{\sum_{s' \in S}} \mathbf{M}_{l_n}'(s_1, s') \mathbin{\widehat{\cdot}} \mathbf{v}_n(s') \\
&= \widehat{\sum_{C \in \mathcal{S}/\mathcal{R}}} \widehat{\sum_{s' \in C}} \mathbf{M}_{l_n}'(s_1, s') \mathbin{\widehat{\cdot}} \mathbf{v}_n(s') \\
&= \widehat{\sum_{C \in \mathcal{S}/\mathcal{R}}} \widehat{\sum_{s' \in C}} T'(s_1, l_n, s') \mathbin{\widehat{\cdot}} \mathbf{v}_n(s') \\
&= \widehat{\sum_{C \in \mathcal{S}/\mathcal{R}}} \widehat{\sum_{s' \in C}} T'(s_2, l_n, s') \mathbin{\widehat{\cdot}} \mathbf{v}_n(s')
\end{aligned}$$

$$= \widehat{\sum_{s' \in S}} \mathbf{M}'_{l_n}(s_2, s') \mathbin{\widehat{\cdot}} \mathbf{v}_n(s')$$
$$= \mathbf{v}_{n-1}(s_2).$$

Note that the equality between the 3rd and 4th line requires that $T'(s_1, l_n, C) = T'(s_2, l_n, C)$ and that all $s' \in C$ have the same value $\mathbf{v}_n(s')$, which is the case here.

We apply the same argumentation to vectors $\mathbf{v}_{n-2}, \mathbf{v}_{n-3}, \ldots, \mathbf{v}_0$, and see that for any equivalence class C, all $s \in C$ have the same value $\mathbf{v}_0(s)$.

So $\widehat{\sum}_{C \in \mathcal{S}/\mathcal{R}} \widehat{\sum}_{s \in C_1} \alpha_1(s) \mathbin{\widehat{\cdot}} \mathbf{v}_0(s) = \widehat{\sum}_{C \in \mathcal{S}/\mathcal{R}} \widehat{\sum}_{s \in C_2} \alpha_2(s) \mathbin{\widehat{\cdot}} \mathbf{v}_0(s)$ holds, if $\alpha_1(C_1) = \alpha_2(C_2)$, which is the case by definition of $\mathcal{A}_1 \approx \mathcal{A}_2$. Hence for $l_1, \ldots, l_n$ holds $w_1^*(\hat{\sigma}) = w_2^*(\hat{\sigma})$. □

The following lemma shows that compositions of different weak bisimulations yield new bisimulations such that new bisimulations can be constructed from known bisimulations.

Lemma 3.1 *If $\mathcal{R}_1, \mathcal{R}_2 \subseteq \mathcal{S} \times \mathcal{S}$ are weak bisimulations for some automaton $\mathcal{A}$, then the following relations are all weak bisimulations:*

1. id *which is the identity,*
2. $\mathcal{R}_1 \cup^* \mathcal{R}_2$*,*

where $\cup^*$ is defined to be the transitive closure of $\mathcal{R}_1 \cup \mathcal{R}_2$.

Proof. We prove 2, the proof of 1 is trivial. Let $\mathcal{R}_0 = \mathcal{R}_1 \cup^* \mathcal{R}_2$, which is an equivalence relation since $\mathcal{R}_1, \mathcal{R}_2$ are equivalence relations. Let $(s, s') \in \mathcal{R}_0$, then either

1. $(s, s') \in \mathcal{R}_i$ $(i = 1, 2)$, or
2. states $s_0, \ldots, s_K$ exist such that $s = s_0$, $s' = s_K$ and $(s_{k-1}, s_k) \in \mathcal{R}_{i_k}$ $(1 < k \leq K$ and $i_k \in \{1, 2\})$.

For a weak bisimulation, we have to show that $\alpha(s) = \alpha(s')$, $\beta'(s) = \beta'(s')$ and $T'(s_1, l, C) = T'(s_2, l, C)$. In Case 1, the conditions are observed for s and s' since $\mathcal{R}_1$ and $\mathcal{R}_2$ are weak bisimulations. In Case 2, the conditions are observed for each pair s_{k-1}, s_k and consequently also for s_0, s_K. □

Theorem 3.2 *For any finite $\mathbb{K}i$-automaton, a largest weak bisimulation exists that is unique up to renumbering of equivalence classes.*

The largest weak bisimulation is the one that has the smallest number of equivalence classes of all weak bisimulations of the $\mathbb{K}i$-automaton.

Proof. Using the previous lemma, $\approx$ is the union of all weak bisimulations of the automaton. □

Clearly, the largest weak bisimulation is the most relevant, because it contains the smallest number of equivalence classes and it can be used to generate the smallest

equivalent automaton by substituting each equivalence class by a single state. The smallest equivalent automaton is unique up to the ordering of states.

Lemma 3.1 gives a characterization of weak bisimulation that characterizes the equivalence classes from below, i. e., small classes are combined to yield larger classes. For the development of algorithms to compute the largest bisimulation for a given automaton or to decide weak bisimilarity of automata, it is often more appropriate to characterize weak bisimulation from above. In this case, a coarse equivalence relation is refined until it is a weak bisimulation. Then computation of the largest weak bisimulation corresponds to a fixed point computation.

Definition 3.2 *Define a function $\mathcal{F} : \mathcal{R} \to \mathcal{R}$ over equivalence relations $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$ as follows: $(s_1, s_2) \in \mathcal{F}(\mathcal{R})$ iff $\alpha(s_1) = \alpha(s_2)$, $\beta'(s_1) = \beta'(s_2)$, and $T'(s_1, l, C) = T'(s_2, l, C)$ holds for all $l \in \mathcal{L}'$ and all equivalence classes $C \in \mathcal{S}/\mathcal{R}$.*

Lemma 3.2 1. *$\mathcal{F}$ is monotonic; that is, if $\mathcal{R}_1 \subseteq \mathcal{R}_2$, then $\mathcal{F}(\mathcal{R}_1) \subseteq \mathcal{F}(\mathcal{R}_2)$,*

2. *$\mathcal{R}$ is a weak bisimulation iff $\mathcal{R} \subseteq \mathcal{F}(\mathcal{R})$.*

Proof. 1. follows directly from the definition of $\mathcal{F}(\mathcal{R})$. 2. is a simple reformulation of the definition of weak bisimulation. □

According to [21], we can formulate the following lemma to characterize $\approx$.

Lemma 3.3 *Relation $\approx$ is the largest fixed point of $\mathcal{F}$.*

Proof. The proof follows the proof of Milner [21, Proposition 16]. □

The previous lemma introduces a simple framework for an algorithm to compute relation $\approx$. Start with relation $\mathcal{R}$ such that $(s_1, s_2) \in \mathcal{R}$ for all $s_1, s_2 \in \mathcal{R}$ and compute $\mathcal{F}(\mathcal{R})$ until the fixed point has been reached. For finite automata, this process terminates after finitely many steps. Moreover, relation $\approx$ is computed with an effort in $O(nm)$, where n is the number of states and m is the number of transitions. The corresponding algorithm for the boolean semiring is described in [10], its extension to the semiring of real numbers with $+$ and $\cdot$ is proposed in [7]. An extension to more general semirings is straightforward. Whether more efficient algorithms like the one for the boolean semiring [12, 24] with an effort in $O(n \log m)$ can be extended to more general semirings and whether such an extension yields a more efficient algorithm for practical applications is an open problem. Before $\approx$ can be computed, $\mathcal{A}^*$ has to be generated from $\mathcal{A}$. If this requires an effort in $O(n^3)$ then the time is dominated by this computation. However, as already mentioned at the end of section 2, $\mathcal{A}^*$ can often be generated more efficiently such that an efficient generation of $\approx$ becomes important.

4. Composition of Weighted Automata

The notion of weak bisimulation is of great value, if it is a congruence. In that case, automata can be replaced by equivalent ones that have smaller sets of states to achieve

minimal representations. We first introduce some composition operations namely direct sum, direct product and synchronized product for automata and show afterwards the congruence property. For instance, direct sum and product are introduced in [9] for stochastic automata. Since the direct product allows one to compose automata sequentially, one may denote it as cascading product as well. Our definition of direct product differs slightly from the one in [9] that merges automata by replicating and redirecting existing transitions while our definition connects automata by additional τ-labelled transitions. We believe that our definition is as natural and useful as the other one but in addition makes the bisimulation a congruence. Free and synchronous product are considered in [2].

Definition 4.1 (Composition of $\mathbb{K}i$-automata) *Let* $\mathcal{A}_1 = (\mathcal{S}_1, \alpha_1, T_1, \beta_1)$ *and* $\mathcal{A}_2 = (\mathcal{S}_2, \alpha_2, T_2, \beta_2)$ *be two finite* $\mathbb{K}i$*-automata.*

The direct sum *of automata is defined as an automaton* $\mathcal{A} = \mathcal{A}_1 + \mathcal{A}_2$ *with*

$$\begin{aligned}
&\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2,\ \mathcal{L} = \mathcal{L}_1 \cup \mathcal{L}_2,\\
&\alpha(s) = \alpha_1(s) \text{ if } s \in \mathcal{S}_1 \text{ and } \alpha(s) = \alpha_2(s) \text{ otherwise,}\\
&\beta(s) = \beta_1(s) \text{ if } s \in \mathcal{S}_1 \text{ and } \beta(s) = \beta_2(s) \text{ otherwise, and}\\
&T(s,l,s') = \begin{cases} T_1(s,l,s') & \text{if } s,s' \in \mathcal{S}_1,\\ T_2(s,l,s') & \text{if } s,s' \in \mathcal{S}_2,\\ \mathbb{O} & \text{otherwise.}\end{cases}
\end{aligned}$$

The direct product *of automata is defined as an automaton* $\mathcal{A} = \mathcal{A}_1 \cdot \mathcal{A}_2$ *with*

$$\begin{aligned}
&\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2,\ \mathcal{L} = \mathcal{L}_1 \cup \mathcal{L}_2,\\
&\alpha(s) = \alpha_1(s) \text{ if } s \in \mathcal{S}_1 \text{ and } \alpha(s) = \mathbb{O} \text{ otherwise,}\\
&\beta(s) = \beta_2(s) \text{ if } s \in \mathcal{S}_2 \text{ and } \beta(s) = \mathbb{O} \text{ otherwise,}\\
&T(s,l,s') = \begin{cases} T_1(s,l,s') & \text{if } s,s' \in \mathcal{S}_1,\\ T_2(s,l,s') & \text{if } s,s' \in \mathcal{S}_2,\\ \beta_1(s) \mathbin{\hat{\cdot}} \alpha_2(s') & \text{if } s \in \mathcal{S}_1,\ s' \in \mathcal{S}_2,\ l = \tau,\\ \mathbb{O} & \text{otherwise.}\end{cases}
\end{aligned}$$

The synchronized product over a set $\mathcal{L}_c \subseteq \{\mathcal{L}_1 \cup \mathcal{L}_2\} \setminus \{\tau\}$ *is defined as an automaton* $\mathcal{A} = \mathcal{A}_1 ||_{\mathcal{L}_c} \mathcal{A}_2$ *over a label set* $\mathcal{L} = \mathcal{L}_1 \cup \mathcal{L}_2$ *and state space* $\mathcal{S} = \mathcal{S}_1 \times \mathcal{S}_2$*. Let* $s = (s_1, s_2)$ *and* $s' = (s'_1, s'_2)$*,* $\alpha(s) = \alpha(s_1) \mathbin{\hat{\cdot}} \alpha(s_2)$ *,* $\beta(s) = \beta(s_1) \mathbin{\hat{\cdot}} \beta(s_2)$*, and*

$$T(s,l,s') = \begin{cases} T_1(s_1,l,s'_1) & \text{if } l \in \mathcal{L}_1 \setminus \{\mathcal{L}_c\},\ s_1 \neq s'_1,\ s_2 = s'_2,\\ T_2(s_2,l,s'_2) & \text{if } l \in \mathcal{L}_2 \setminus \{\mathcal{L}_c\},\ s_1 = s'_1,\ s_2 \neq s'_2,\\ T_1(s_1,l,s'_1) \mathbin{\hat{+}} T_2(s_2,l,s'_2) & \text{if } l \in (\mathcal{L}_1 \cup \mathcal{L}_2) \setminus \{\mathcal{L}_c\},\ s = s',\\ T_1(s_1,l,s'_1) \mathbin{\hat{\cdot}} T_2(s_2,l,s'_2) & \text{if } l \in \mathcal{L}_c,\\ \mathbb{O} & \text{otherwise.}\end{cases}$$

The free product results from a synchronous product with $\mathcal{L}_c = \emptyset$*.*

The composition operations are defined by means of the functions T, α and β. Note that the role of α and β in the direct product is to indicate states that can be used to connect automata. $\beta(s) \neq \mathbb{O}$ indicates possible end states of $\mathcal{A}_1$ to which starting states ($\alpha(s) \neq \mathbb{O}$) of $\mathcal{A}_2$ can be connected. It is also possible to influence the weight of a path through a connection between $\mathcal{A}_1$ and $\mathcal{A}_2$ by the chosen element of $\mathbb{K}i$. However, this is an option that need not be used; $\mathbb{1}$ and $\mathbb{O}$ as boolean indicators are sufficient to allow or deny a connection.

Alternatively, one can define the operations using vectors and matrices as follows. Let $\mathbf{a}^{(i)}$, $\mathbf{b}^{(i)}$ and $\mathbf{M}_l^{(i)}$ $(i = 1, 2)$ be the vectors and matrices describing automata $\mathcal{A}_1$ and $\mathcal{A}_2$. Let $\mathbb{O}$ also denote a vector or matrix with all elements $\mathbb{O}$. The representation of $\mathcal{A} = \mathcal{A}_1 + \mathcal{A}_2$ in terms of vectors and matrices is given by

$$\mathbf{a} = \big(\mathbf{a}^{(1)}, \mathbf{a}^{(2)}\big), \quad \mathbf{b} = \begin{pmatrix} \mathbf{b}^{(1)} \\ \mathbf{b}^{(2)} \end{pmatrix}, \quad \mathbf{M}_l = \begin{pmatrix} \mathbf{M}_l^{(1)} & \mathbb{O} \\ \mathbb{O} & \mathbf{M}_l^{(2)} \end{pmatrix}.$$

where $\mathbf{b}^{(1)}$, $\mathbf{b}^{(2)}$ denote column vectors, $\mathbf{a}^{(1)}$, $\mathbf{a}^{(2)}$ row vectors. For $\mathcal{A} = \mathcal{A}_1 || _{\mathcal{L}_c} \mathcal{A}_2$, we obtain the following representation.

$$\mathbf{a} = \mathbf{a}^{(1)} \mathbin{\widehat{\otimes}} \mathbf{a}^{(2)}, \quad \mathbf{b} = \mathbf{b}^{(1)} \mathbin{\widehat{\otimes}} \mathbf{b}^{(2)},$$

$$\mathbf{M}_l = \mathbf{M}_l^{(1)} \mathbin{\widehat{\otimes}} \mathbf{M}_l^{(2)} \text{ if } l \in \mathcal{L}_c \text{ and } \mathbf{M}_l = \mathbf{M}_l^{(1)} \mathbin{\widehat{\oplus}} \mathbf{M}_l^{(2)} \text{ otherwise.}$$

Matrices for $\mathcal{A} = \mathcal{A}_1 \cdot \mathcal{A}_2$ are defined as

$$\mathbf{a} = \big(\mathbf{a}^{(1)}, \mathbb{O}\big), \quad \mathbf{b} = \begin{pmatrix} \mathbb{O} \\ \mathbf{b}^{(2)} \end{pmatrix},$$

$$\mathbf{M}_l = \begin{pmatrix} \mathbf{M}_l^{(1)} & \mathbb{O} \\ \mathbb{O} & \mathbf{M}_l^{(2)} \end{pmatrix} (l \neq \tau) \quad \text{and} \quad \mathbf{M}_\tau = \begin{pmatrix} \mathbf{M}_\tau^{(1)} & \mathbf{b}^{(1)}\mathbf{a}^{(2)} \\ \mathbb{O} & \mathbf{M}_\tau^{(2)} \end{pmatrix}.$$

It is straightforward to prove that both definitions of the composition operations are identical. Furthermore, the direct product $\mathcal{A} = \mathcal{A}_1 \cdot \mathcal{A}_2$ also has a concise matrix description of $\mathbf{M}_\tau^*$ and $\mathbf{M}_l'$.

$$\mathbf{M}_\tau^* = \begin{pmatrix} \mathbf{M}_\tau^{(1)*} & \mathbf{M}_\tau^{(1)*}\mathbf{b}^{(1)}\mathbf{a}^{(2)}\mathbf{M}_\tau^{(2)*} \\ \mathbb{O} & \mathbf{M}_\tau^{(2)*} \end{pmatrix}, \tag{8}$$

$$\mathbf{M}_l' = \mathbf{M}_\tau^*\mathbf{M}_l\mathbf{M}_\tau^* = \begin{pmatrix} \mathbf{M}_l^{(1)'} & \mathbf{M}_l^{(1)'}\mathbf{b}^{(1)}\mathbf{a}^{(2)}\mathbf{M}_\tau^{(2)*} \mathbin{\widehat{+}} \mathbf{M}_\tau^{(1)*}\mathbf{b}^{(1)}\mathbf{a}^{(2)}\mathbf{M}_l^{(2)'} \\ \mathbb{O} & \mathbf{M}_l^{(2)'} \end{pmatrix}. \tag{9}$$

where $\mathbf{M}_l^{(i)'} = \mathbb{O}$ if l is not present in automaton $\mathcal{A}_i$ for $i \in \{1, 2\}$. Matrix $\mathbf{M}_l'$ covers both cases, namely that l is present in the first or the second automaton. The result follows directly from $\mathbf{M}_\tau^*\mathbf{M}_l\mathbf{M}_\tau^*$. Matrix $\mathbf{M}_\tau^*$ is a special case of the following lemma.

Lemma 4.1 *Let* $\mathbf{M} \in \mathbb{K}i^{n\times n}$ *be a block-structured matrix with* $\mathbf{M} = \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbb{O} & \mathbf{D} \end{pmatrix}$ *then for* $i \in \mathbb{N}$, $i > 0$ *holds*

$$\mathbf{M}^i = \begin{pmatrix} \mathbf{A}^i & \widehat{\sum}_{j=0}^{i-1} \mathbf{A}^j\mathbf{B}\mathbf{D}^{(i-1)-j} \\ \mathbb{O} & \mathbf{D}^i \end{pmatrix} \quad \textit{and} \quad \mathbf{M}^* = \begin{pmatrix} \mathbf{A}^* & \mathbf{A}^*\mathbf{B}\mathbf{D}^* \\ \mathbb{O} & \mathbf{D}^* \end{pmatrix}.$$

Proof. We begin with $\mathbf{M}^i$ and prove it by induction on i. For $i = 1$,

$$\mathbf{M}^1 = \begin{pmatrix} \mathbf{A}^1 & \widehat{\sum}_{j=0}^{0} \mathbf{A}^j\mathbf{B}\mathbf{D}^{0-j} \\ \mathbb{O} & \mathbf{D}^1 \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbb{O} & \mathbf{D} \end{pmatrix}$$

since $\mathbf{A}^0, \mathbf{D}^0$ are identity matrices. For some $i > 1$,

$$\mathbf{M}^i = \mathbf{M}^{i-1}\mathbf{M} = \begin{pmatrix} \mathbf{A}^{i-1} & \widehat{\sum}_{j=0}^{i-2} \mathbf{A}^j\mathbf{B}\mathbf{D}^{(i-2)-j} \\ \mathbb{O} & \mathbf{D}^{i-1} \end{pmatrix} \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbb{O} & \mathbf{D} \end{pmatrix} = \begin{pmatrix} \mathbf{A}^i & \mathbf{B}_i \\ \mathbb{O} & \mathbf{D}^i \end{pmatrix}$$

with $\mathbf{B}_i = \mathbf{A}^{i-1}\mathbf{B} \mathbin{\widehat{+}} \widehat{\sum}_{j=0}^{i-2} \mathbf{A}^j\mathbf{B}\mathbf{D}^{(i-2)-j}\mathbf{D}$ by induction assumption and matrix multiplication. Obviously, $\mathbf{B}_i = \mathbf{A}^{i-1}\mathbf{B}\mathbf{D}^0 \mathbin{\widehat{+}} \widehat{\sum}_{j=0}^{i-2} \mathbf{A}^j\mathbf{B}\mathbf{D}^{(i-1)-j}$. Since $\mathbf{A}^{i-1}\mathbf{B}\mathbf{D}^0 = \mathbf{A}^j\mathbf{B}\mathbf{D}^{(i-1)-j}$ for $j = i-1$, we obtain $\mathbf{B}_i = \widehat{\sum}_{j=0}^{i-1} \mathbf{A}^j\mathbf{B}\mathbf{D}^{(i-1)-j}$ and complete the induction step. It remains to consider $\mathbf{M}^*$. By definition $\mathbf{M}^* = \widehat{\sum}_{i=0}^{\infty} \mathbf{M}^i$. By help of the aforegoing result and $\mathbf{M}^0 = \mathbf{I}$ we obtain

$$\begin{aligned} \mathbf{M}^* &= \mathbf{I} \mathbin{\widehat{+}} \widehat{\sum_{i=1}^{\infty}} \begin{pmatrix} \mathbf{A}^i & \widehat{\sum}_{j=0}^{i-1} \mathbf{A}^j\mathbf{B}\mathbf{D}^{(i-1)-j} \\ \mathbb{O} & \mathbf{D}^i \end{pmatrix} \\ &= \begin{pmatrix} \widehat{\sum}_{i=0}^{\infty} \mathbf{A}^i & \widehat{\sum}_{i=1}^{\infty} \widehat{\sum}_{j=0}^{i-1} \mathbf{A}^j\mathbf{B}\mathbf{D}^{(i-1)-j} \\ \mathbb{O} & \widehat{\sum}_{i=0}^{\infty} \mathbf{D}^i \end{pmatrix} = \begin{pmatrix} \mathbf{A}^* & \mathbf{B}_* \\ \mathbb{O} & \mathbf{D}^* \end{pmatrix} \end{aligned}$$

with $\mathbf{B}_* = \widehat{\sum}_{i=1}^{\infty} \widehat{\sum}_{j=0}^{i-1} \mathbf{A}^j\mathbf{B}\mathbf{D}^{(i-1)-j}$. By reordering the terms in the summation, we obtain

$$\mathbf{B}_* = \widehat{\sum_{j=0}^{\infty}} \widehat{\sum_{i=0}^{\infty}} \mathbf{A}^j\mathbf{B}\mathbf{D}^i = \widehat{\sum_{j=0}^{\infty}} \mathbf{A}^j\mathbf{B} \widehat{\sum_{i=0}^{\infty}} \mathbf{D}^i = \widehat{\sum_{j=0}^{\infty}} \mathbf{A}^j\mathbf{B}\mathbf{D}^* = \mathbf{A}^*\mathbf{B}\mathbf{D}^*,$$

which gives the desired result. □

The lemma allows us to immediately obtain $\mathbf{M}_\tau^*$ for the direct product of automata.

The following theorems introduce the main result of this section, namely the congruence property of weak bisimulation according to the composition operations.

Theorem 4.1 *If $\mathcal{A}_1 \approx \mathcal{A}_2$ and $\mathcal{A}_3$ are finite $\mathbb{K}i$-automata, then*

1. $\mathcal{A}_1 + \mathcal{A}_3 \approx \mathcal{A}_2 + \mathcal{A}_3$,
2. $\mathcal{A}_1 \cdot \mathcal{A}_3 \approx \mathcal{A}_2 \cdot \mathcal{A}_3$,
3. $\mathcal{A}_3 \cdot \mathcal{A}_1 \approx \mathcal{A}_3 \cdot \mathcal{A}_2$.

Proof. In the following proofs, $\mathcal{A}_1 \approx \mathcal{A}_2$ holds and $\mathcal{R}_{12}$ is a weak bisimulation relation assuring $\approx$ according to Def. 3.1. We define a relation by $\mathcal{R}_{123} \subseteq S_{123} \times S_{123}$ with $S_{123} = \mathcal{S}_1 \cup \mathcal{S}_2 \cup \mathcal{S}_3$ and $(s, s') \in \mathcal{R}_{123}$ iff $(s, s') \in \mathcal{R}_{12}$ or $s = s'$, i. e., $\mathcal{R}_{123} = \mathcal{R}_{12} \cup^* \mathrm{id} = \mathcal{R}_{12} \cup \mathrm{id}$.

The fact that $\mathcal{R}_{123}$ is an equivalence relation is straightforward to prove. We prove that $\mathcal{R}_{123}$ is a weak bisimulation for the different composition operations, i. e.,

$\forall(s_1,s_2)\in\mathcal{R}_{123}:(\alpha(s_1)=\alpha(s_2))\wedge(\beta'(s_1)=\beta'(s_2))\wedge(T'(s_1,l,C)=T'(s_2,l,C))$ for all equivalence classes $C\in\mathcal{S}_{123}/\mathcal{R}_{123}$. Let $(s_1,s_2)\in\mathcal{R}_{123}$.

Direct sum: If $(s_1,s_2)\in\mathcal{R}_{12}$, then the conditions are satisfied because $\mathcal{R}_{12}$ is a weak bisimulation. If $(s_1,s_2)\notin\mathcal{R}_{12}$, then $s_1=s_2$ and the conditions are trivially fulfilled.

Direct product, first part: The condition $\alpha(s_1)=\alpha(s_2)$ for all $(s_1,s_2)\in\mathcal{R}_{123}$ follows directly from the definition of $\mathcal{R}_{123}$. If $s_1,s_2\in\mathcal{S}_1\cup\mathcal{S}_2$, then the condition is observed since $\mathcal{R}_{12}$ is a weak bisimulation. In the remaining cases, $s_1=s_2$ and the condition is trivially observed.

By definition of direct product, (8), and (9) we obtain matrices

$$\mathbf{M}_\tau^{(13)*}=\begin{pmatrix}\mathbf{M}_\tau^{(1)*} & \mathbf{M}_\tau^{(1)*}\mathbf{b}^{(1)}\mathbf{a}^{(3)}\mathbf{M}_\tau^{(3)*}\\ \mathbb{O} & \mathbf{M}_\tau^{(3)*}\end{pmatrix},$$

$$\mathbf{M}_\tau^{(23)*}=\begin{pmatrix}\mathbf{M}_\tau^{(2)*} & \mathbf{M}_\tau^{(2)*}\mathbf{b}^{(2)}\mathbf{a}^{(3)}\mathbf{M}_\tau^{(3)*}\\ \mathbb{O} & \mathbf{M}_\tau^{(3)*}\end{pmatrix},$$

$$\mathbf{M}_l^{(13)'}=\begin{pmatrix}\mathbf{M}_l^{(1)'} & \mathbf{M}_l^{(1)'}\mathbf{b}^{(1)}\mathbf{a}^{(3)}\mathbf{M}_\tau^{(3)*}+\mathbf{M}_\tau^{(1)*}\mathbf{b}^{(1)}\mathbf{a}^{(3)}\mathbf{M}_l^{(3)'}\\ \mathbb{O} & \mathbf{M}_l^{(3)'}\end{pmatrix},$$

$$\mathbf{M}_l^{(23)'}=\begin{pmatrix}\mathbf{M}_l^{(2)'} & \mathbf{M}_l^{(2)'}\mathbf{b}^{(2)}\mathbf{a}^{(3)}\mathbf{M}_\tau^{(3)*}+\mathbf{M}_\tau^{(2)*}\mathbf{b}^{(2)}\mathbf{a}^{(3)}\mathbf{M}_l^{(3)'}\\ \mathbb{O} & \mathbf{M}_l^{(3)'}\end{pmatrix}.$$

Let $\mathbf{e}_S^i$ be row a vector of length $\mathcal{S}_i$ for $i\in\{1,2,3\}$ and $\mathbf{e}_S^i(s)=\mathbb{1}$ if $s\in S\cap\mathcal{S}_i$ and $\mathbb{O}$ otherwise. For a singleton $S=\{s\}$, we write $\mathbf{e}_s$ and omit superscript i if it is clear which $\mathcal{S}_i$ is considered.

Condition $\forall(s_1,s_2)\in\mathcal{R}_{123}:\beta'(s_1)=\beta'(s_2)$ is observed for $(s_1,s_2)\in\mathcal{R}_{12}$ since

$$\begin{aligned}\beta'(s_1)&=(\mathbf{e}_{s_1},\mathbb{O})\mathbin{\widehat{\cdot}}\mathbf{M}_\tau^{(13)*}\mathbin{\widehat{\cdot}}\begin{pmatrix}\mathbb{O}\\ \mathbf{b}^{(3)}\end{pmatrix}\\ &=\mathbf{e}_{s_1}\mathbin{\widehat{\cdot}}\mathbf{M}_\tau^{(1)*}\mathbin{\widehat{\cdot}}\mathbf{b}^{(1)}\mathbin{\widehat{\cdot}}\mathbf{a}^{(3)}\mathbin{\widehat{\cdot}}\mathbf{M}_\tau^{(3)*}\mathbin{\widehat{\cdot}}\mathbf{b}^{(3)}\\ &=\beta_1'(s_1)\mathbin{\widehat{\cdot}}\mathbf{a}^{(3)}\mathbin{\widehat{\cdot}}\mathbf{M}_\tau^{(3)*}\mathbin{\widehat{\cdot}}\mathbf{b}^{(3)}\\ &=\beta_2'(s_2)\mathbin{\widehat{\cdot}}\mathbf{a}^{(3)}\mathbin{\widehat{\cdot}}\mathbf{M}_\tau^{(3)*}\mathbin{\widehat{\cdot}}\mathbf{b}^{(3)}\\ &=\mathbf{e}_{s_2}\mathbin{\widehat{\cdot}}\mathbf{M}_\tau^{(2)*}\mathbin{\widehat{\cdot}}\mathbf{b}^{(2)}\mathbin{\widehat{\cdot}}\mathbf{a}^{(3)}\mathbin{\widehat{\cdot}}\mathbf{M}_\tau^{(3)*}\mathbin{\widehat{\cdot}}\mathbf{b}^{(3)}\\ &=(\mathbf{e}_{s_2},\mathbb{O})\mathbin{\widehat{\cdot}}\mathbf{M}_\tau^{(23)*}\mathbin{\widehat{\cdot}}\begin{pmatrix}\mathbb{O}\\ \mathbf{b}^{(3)}\end{pmatrix}\\ &=\beta'(s_2)\end{aligned}$$

for $s_1,s_2\in\mathcal{S}_1\cup\mathcal{S}_2$ where β_i' $(i=1,2)$ denotes function β' in automaton i and β' is the function in the automaton resulting from composition. For $s_1,s_2\in\mathcal{S}_3$, $s_1=s_2$ has to hold by definition of $\mathcal{R}_{123}$ and $\beta'(s_1)=\beta'(s_2)$ is immediate.

It remains to show the condition on the transition weights, i. e. $T'(s_1,l,C)=T'(s_2,l,C)$ for any $s_1\approx s_2$ and $C\in\mathcal{S}_{123}/\mathcal{R}_{123}$. By definition of $\mathcal{S}_{123}/\mathcal{R}_{123}$, C is either a subset of $\mathcal{S}_3$ or $\mathcal{S}_1\cup\mathcal{S}_2$, such that we consider four cases.

Case 1: $s_1, s_2 \in \mathcal{S}_1 \cup \mathcal{S}_2$ and $C \subseteq \mathcal{S}_1 \cup \mathcal{S}_2$.
This case retains within $\mathcal{A}_1, \mathcal{A}_2$ and holds by $(s_1, s_2) \in \mathcal{R}_{12}$. Formally, for $l = \epsilon$,

$$T'(s_1, l, C) = (\mathbf{e}_{s_1}, \mathbb{O})\mathbf{M}_\tau^{(13)*}(\mathbf{e}_C^1, \mathbb{O})^T = \mathbf{e}_{s_1}\mathbf{M}_\tau^{(1)*}(\mathbf{e}_C^1)^T \text{ and}$$
$$T'(s_2, l, C) = (\mathbf{e}_{s_2}, \mathbb{O})\mathbf{M}_\tau^{(23)*}(\mathbf{e}_C^2, \mathbb{O})^T = \mathbf{e}_{s_2}\mathbf{M}_\tau^{(2)*}(\mathbf{e}_C^2)^T$$

are equal due to $(s_1, s_2) \in \mathcal{R}_{12}$.

For $l \neq \epsilon$,

$$T'(s_1, l, C) = (\mathbf{e}_{s_1}, \mathbb{O})\mathbf{M}_l^{(13)'}(\mathbf{e}_C^1, \mathbb{O})^T = \mathbf{e}_{s_1}\mathbf{M}_l^{(1)'}(\mathbf{e}_C^1)^T \text{ and}$$
$$T'(s_2, l, C) = (\mathbf{e}_{s_2}, \mathbb{O})\mathbf{M}_l^{(23)'}(\mathbf{e}_C^2, \mathbb{O})^T = \mathbf{e}_{s_2}\mathbf{M}_l^{(2)'}(\mathbf{e}_C^2)^T$$

are equal due to $s_1, s_2 \in \mathcal{R}_{12}$.

Case 2: $s_1, s_2 \in \mathcal{S}_1 \cup \mathcal{S}_2$ and $C \subseteq \mathcal{S}_3$.
Due to $\mathcal{R}_{123}$, C contains only a single state which is denoted by $s_3 \in \mathcal{S}_3$. For $l = \epsilon$, we have

$$\begin{aligned}
T'(s_1, \epsilon, s_3) &= (\mathbf{e}_{s_1}, \mathbb{O}) \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(13)*} \mathbin{\widehat{\cdot}} (\mathbb{O}, \mathbf{e}_{s_3})^T \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(1)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(1)} \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} (\mathbf{e}_{s_3})^T \\
&= \beta_1'(s_1) \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} (\mathbf{e}_{s_3})^T \\
&= \beta_2'(s_2) \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} (\mathbf{e}_{s_3})^T \\
&= \mathbf{e}_{s_2} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(2)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(2)} \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} (\mathbf{e}_{s_3})^T \\
&= (\mathbf{e}_{s_2}, \mathbb{O}) \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(23)*} \mathbin{\widehat{\cdot}} (\mathbb{O}, \mathbf{e}_{s_3})^T \\
&= T'(s_2, \epsilon, s_3).
\end{aligned}$$

For $l \neq \epsilon$, we distinguish two cases, namely the occurrence of l in $\mathcal{A}_1$ or $\mathcal{A}_2$ and the occurrence of l in $\mathcal{A}_3$. This simplifies matrices $\mathbf{M}_l^{(13)'}$ and $\mathbf{M}_l^{(23)'}$ accordingly. We start with the former case and have

$$\begin{aligned}
T'(s_1, l, s_3) &= (\mathbf{e}_{s_1}, \mathbb{O}) \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(13)'} \mathbin{\widehat{\cdot}} (\mathbb{O}, \mathbf{e}_{s_3})^T \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(1)'} \mathbin{\widehat{\cdot}} \mathbf{b}^{(1)} \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*}(\mathbf{e}_{s_3})^T \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(1)'} \mathbin{\widehat{\cdot}} \mathbf{b}^{(1)'} \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*}(\mathbf{e}_{s_3})^T.
\end{aligned}$$

We can replace $\mathbf{b}^{(1)}$ by $\mathbf{b}^{(1)'}$ above because idempotency of the semiring ensures that $\mathbf{M}_\tau^{(1)*} = \mathbf{M}_\tau^{(1)*}\mathbf{M}_\tau^{(1)*}$ such that $\mathbf{M}_l^{(1)'} \mathbin{\widehat{\cdot}} \mathbf{b}^{(1)} = \mathbf{M}_\tau^{(1)*}\mathbf{M}_l^{(1)}\mathbf{M}_\tau^{(1)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(1)} = \mathbf{M}_\tau^{(1)*}\mathbf{M}_l^{(1)}\mathbf{M}_\tau^{(1)*}\mathbf{M}_\tau^{(1)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(1)} = \mathbf{M}_l^{(1)'} \mathbin{\widehat{\cdot}} \mathbf{b}^{(1)'}$. We can distinguish among the different classes $C' \in \mathcal{S}/\mathcal{R}_{12}$, let $\mathbf{b}_{C'}^{(i)}$ be a vector of length $|\mathcal{S}_i|$ with $\mathbf{b}^{(i)}(s)$ in position $s \in \mathcal{S}_i$ if $s \in C' \cap \mathcal{S}_i$, otherwise the value in position s is $\mathbb{O}$. Consequently, $\mathbf{b}^{(i)} = \widehat{\sum}_{C' \in \mathcal{S}/\mathcal{R}_{12}} \mathbf{b}_{C'}^{(i)}$. We exploit that $c' = \mathbf{b}_{C'}^{(1)}(s) = \mathbf{b}_{C'}^{(2)}(s')$ is the same for any $s \in C' \cap \mathcal{S}_1$, $s' \in C' \cap \mathcal{S}_2$. Now, let $\mathbf{x}^{(1)} = \mathbf{M}_l^{(1)'} \mathbin{\widehat{\cdot}} (\mathbf{e}_{C'}^1)^T$ and $\mathbf{x}^{(2)} = \mathbf{M}_l^{(2)'} \mathbin{\widehat{\cdot}} (\mathbf{e}_{C'}^2)^T$, then $\mathcal{R}_{12}$ ensures that $\mathbf{x}^{(1)}(s) = \mathbf{x}^{(2)}(s')$ for any $(s, s') \in \mathcal{R}_{12}$. Note that $\mathbf{x}^{(1)}$, $\mathbf{x}^{(2)}$

and c' depend on the considered C'.

$$\begin{aligned}
T'(s_1,l,s_3) &= \left(\widehat{\sum_{C'\in\mathcal{S}/\mathcal{R}_{12}}} \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(1)'} \mathbin{\widehat{\cdot}} \mathbf{b}_{C'}^{(1)}\right) \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*}(\mathbf{e}_{s_3})^T \\
&= \left(\widehat{\sum_{C'\in\mathcal{S}/\mathcal{R}_{12}}} \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(1)'} \mathbin{\widehat{\cdot}} (\mathbf{e}_{C'}^{(1)})^T \mathbin{\widehat{\cdot}} c'\right) \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*}(\mathbf{e}_{s_3})^T \\
&= \left(\widehat{\sum_{C'\in\mathcal{S}/\mathcal{R}_{12}}} \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{x}^{(1)} \mathbin{\widehat{\cdot}} c'\right) \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*}(\mathbf{e}_{s_3})^T \\
&= \left(\widehat{\sum_{C'\in\mathcal{S}/\mathcal{R}_{12}}} \mathbf{e}_{s_2} \mathbin{\widehat{\cdot}} \mathbf{x}^{(2)} \mathbin{\widehat{\cdot}} c'\right) \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*}(\mathbf{e}_{s_3})^T \\
&= \left(\widehat{\sum_{C'\in\mathcal{S}/\mathcal{R}_{12}}} \mathbf{e}_{s_2} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(2)'} \mathbin{\widehat{\cdot}} \mathbf{b}_{C'}^{(2)}\right) \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*}(\mathbf{e}_{s_3})^T \\
&= \mathbf{e}_{s_2} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(2)'} \mathbin{\widehat{\cdot}} \mathbf{b}^{(2)'} \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*}(\mathbf{e}_{s_3})^T \\
&= (\mathbf{e}_{s_2}, \mathbb{O}) \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(23)'} \mathbin{\widehat{\cdot}} (\mathbb{O}, \mathbf{e}_{s_3})^T \\
&= T'(s_2,l,s_3).
\end{aligned}$$

For the case that l is a transition in $\mathcal{A}_3$, we have

$$\begin{aligned}
T'(s_1,l,s_3) &= (\mathbf{e}_{s_1}, \mathbb{O}) \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(13)'} \mathbin{\widehat{\cdot}} (\mathbb{O}, \mathbf{e}_{s_3})^T \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{b}^{(1)'} \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(3)'} \mathbin{\widehat{\cdot}} (\mathbf{e}_{s_3})^T \\
&= \left(\widehat{\sum_{C\in\mathcal{S}/\mathcal{R}_{12}}} \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{b}_C^{(1)'}\right) \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(3)'} \mathbin{\widehat{\cdot}} (\mathbf{e}_{s_3})^T \\
&= \left(\widehat{\sum_{C\in\mathcal{S}/\mathcal{R}_{12}}} \mathbf{e}_{s_2} \mathbin{\widehat{\cdot}} \mathbf{b}_C^{(2)'}\right) \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(3)'} \mathbin{\widehat{\cdot}} (\mathbf{e}_{s_3})^T \\
&= \mathbf{e}_{s_2} \mathbf{b}^{(2)'} \mathbin{\widehat{\cdot}} \mathbf{a}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(3)'} (\mathbf{e}_{s_3})^T \\
&= (\mathbf{e}_{s_2}, \mathbb{O}) \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(23)'} \mathbin{\widehat{\cdot}} (\mathbb{O}, \mathbf{e}_{s_3})^T \\
&= T'(s_2,l,s_3)
\end{aligned}$$

where $\mathbf{b}_C^{(1)'}, \mathbf{b}_C^{(2)'}$ are defined analogously to $\mathbf{b}_C^{(i)}$.

Case 3: $s_1, s_2 \in \mathcal{S}_3$ and $C \subseteq \mathcal{S}_1 \cup \mathcal{S}_2$.
For $l = \epsilon$, $T'(s_1,l,C) = (\mathbb{O}, \mathbf{e}_{s_1}) \mathbf{M}_\tau^{(13)*} (\mathbf{e}_C^1, \mathbb{O})^T = \mathbf{e}_{s_1} \mathbb{O} (\mathbf{e}_C^1)^T = \mathbb{O} = T'(s_2,l,C)$.
For $l \neq \epsilon$, $T'(s_1,l,C) = (\mathbb{O}, \mathbf{e}_{s_1}) \mathbf{M}_l^{(13)'} (\mathbf{e}_C^1, \mathbb{O})^T = \mathbf{e}_{s_1} \mathbb{O} (\mathbf{e}_C^1)^T = \mathbb{O} = T'(s_2,l,C)$.

Case 4: $s_1, s_2 \in \mathcal{S}_3$ and $C \subseteq \mathcal{S}_3$.
Since $s_1 = s_2$ by definition of $\mathcal{R}_{123}$, we obtain for $l = \epsilon$, $T'(s_1,l,C) = (\mathbb{O}, \mathbf{e}_{s_1}) \mathbf{M}_\tau^{(13)*} (\mathbb{O}, \mathbf{e}_C^3)^T = \mathbf{e}_{s_1} \mathbf{M}_\tau^{(3)*} (\mathbf{e}_C^3)^T = \mathbf{e}_{s_2} \mathbf{M}_\tau^{(3)*} (\mathbf{e}_C^3)^T = T'(s_2,l,C)$. For

$l \neq \epsilon$, $T'(s_1, l, C) = (\mathfrak{O}, \mathbf{e}_{s_1})\mathbf{M}_l^{(13)'}(\mathfrak{O}, \mathbf{e}_C^3)^T = \mathbf{e}_{s_1}\mathbf{M}_l^{(3)'}(\mathbf{e}_C^3)^T = \mathbf{e}_{s_2}\mathbf{M}_l^{(3)'}(\mathbf{e}_C^3)^T = T'(s_2, l, C)$.

Direct product, second part: By definition of direct product, we obtain for this construction the matrices

$$\mathbf{M}_\tau^{(31)*} = \begin{pmatrix} \mathbf{M}_\tau^{(3)*} & \mathbf{M}_\tau^{(3)*}\mathbf{b}^{(3)}\mathbf{a}^{(1)}\mathbf{M}_\tau^{(1)*} \\ \mathfrak{O} & \mathbf{M}_\tau^{(1)*} \end{pmatrix},$$

$$\mathbf{M}_\tau^{(32)*} = \begin{pmatrix} \mathbf{M}_\tau^{(3)*} & \mathbf{M}_\tau^{(3)*}\mathbf{b}^{(3)}\mathbf{a}^{(2)}\mathbf{M}_\tau^{(2)*} \\ \mathfrak{O} & \mathbf{M}_\tau^{(2)*} \end{pmatrix},$$

$$\mathbf{M}_l^{(31)'} = \begin{pmatrix} \mathbf{M}_l^{(3)'} & \mathbf{M}_l^{(1)'}\mathbf{b}^{(3)}\mathbf{a}^{(1)}\mathbf{M}_\tau^{(1)*} + \mathbf{M}_\tau^{(3)*}\mathbf{b}^{(3)}\mathbf{a}^{(1)}\mathbf{M}_l^{(1)'} \\ \mathfrak{O} & \mathbf{M}_l^{(1)'} \end{pmatrix},$$

$$\mathbf{M}_l^{(32)'} = \begin{pmatrix} \mathbf{M}_l^{(3)'} & \mathbf{M}_l^{(3)'}\mathbf{b}^{(3}\mathbf{a}^{(2)}\mathbf{M}_\tau^{(2)*} + \mathbf{M}_\tau^{(3)*}\mathbf{b}^{(3)}\mathbf{a}^{(2)}\mathbf{M}_l^{(2)'} \\ \mathfrak{O} & \mathbf{M}_l^{(2)'} \end{pmatrix}.$$

Condition $\forall(s_1, s_2) \in \mathcal{R}_{123} : \alpha(s_1) = \alpha(s_2)$ is observed for $s_1, s_2 \in \mathcal{S}_3$ because in this case $s_1 = s_2$. For $s_1, s_2 \in \mathcal{S}_1 \cup \mathcal{S}_2$, by definition $\alpha(s_1) = \mathbb{O} = \alpha(s_2)$.

We consider the condition for β'. We have $\mathbf{b}^{(3i)'} = \mathbf{M}_\tau^{(3i)*}\mathbf{b}^{(3i)}$ for $i \in \{1, 2\}$, but due to $\mathbf{b}^{(3i)} = (\mathfrak{O}, \mathbf{b}^{(i)})$ the expression gets simpler. For $s_1, s_2 \in \mathcal{S}_1 \cup \mathcal{S}_2$, we obtain $\mathbf{b}^{(31)'}(s_1) = \mathbf{M}_\tau^{(1)*}\mathbf{b}^{(1)}$ and $\mathbf{b}^{(32)'}(s_2) = \mathbf{M}_\tau^{(2)*}\mathbf{b}^{(2)}$ which are equal due to $(s_1, s_2) \in \mathcal{R}_{12}$. For $s_1, s_2 \in \mathcal{S}_3$, we have $s_1 = s_2$, but need to apply the definition and consider the impact of each equivalence class $C \in \mathcal{R}_{12}$:

$$\begin{aligned}
\mathbf{b}^{(31)'}(s_1) &= \mathbf{e}_{s_1}\mathbf{M}_\tau^{(3)*}\mathbf{b}^{(3)}\mathbf{a}^{(1)}\mathbf{M}_\tau^{(1)*}\mathbf{b}^{(1)} \\
&= \mathbf{e}_{s_1}\mathbf{M}_\tau^{(3)*}\mathbf{b}^{(3)} \widehat{\sum_{s\in\mathcal{S}_1}} \mathbf{a}^{(1)}(s) \mathbin{\widehat{\cdot}} \mathbf{b}^{(1)'}(s) \\
&= \mathbf{e}_{s_1}\mathbf{M}_\tau^{(3)*}\mathbf{b}^{(3)} \widehat{\sum_{C\in\mathcal{S}_1/\mathcal{R}_{12}}} \underbrace{\widehat{\sum_{s\in C}} \mathbf{a}^{(1)}(s) \mathbin{\widehat{\cdot}} \mathbf{b}^{(1)'}(s)}_{\substack{\text{where for } s\in C_1,\ s'\in C_2: \\ \mathbf{a}^{(1)}(s)=\mathbf{a}^{(2)}(s'),\ \mathbf{b}^{(1)'}(s)=\mathbf{b}^{(2)'}(s')}} \\
&= \mathbf{e}_{s_1}\mathbf{M}_\tau^{(3)*}\mathbf{b}^{(3)} \widehat{\sum_{C\in\mathcal{S}_2/\mathcal{R}_{12}}} \widehat{\sum_{s\in C}} \mathbf{a}^{(2)}(s) \mathbin{\widehat{\cdot}} \mathbf{b}^{(2)'}(s) \\
&= \mathbf{e}_{s_1}\mathbf{M}_\tau^{(3)*}\mathbf{b}^{(3)} \widehat{\sum_{s\in\mathcal{S}_2}} \mathbf{a}^{(2)}(s) \mathbin{\widehat{\cdot}} \mathbf{b}^{(2)'}(s) \\
&= \mathbf{e}_{s_2}\mathbf{M}_\tau^{(3)*}\mathbf{b}^{(3)}\mathbf{a}^{(2)}\mathbf{M}_\tau^{(2)*}\mathbf{b}^{(2)} = \mathbf{b}^{(32)'}(s_2).
\end{aligned}$$

For the proof of the condition on the path weights, we can proceed similarly to the proof of Theorem 4.1.2.

Case 1: $s_1, s_2 \in \mathcal{S}_1 \cup \mathcal{S}_2$ and $C \subseteq \mathcal{S}_1 \cup \mathcal{S}_2$.
For $l = \epsilon$,

$$T'(s_1, l, C) = (\mathbb{O}, \mathbf{e}_{s_1})\mathbf{M}_\tau^{(31)*}(\mathbb{O}, \mathbf{e}_C^1)^T = \mathbf{e}_{s_1}\mathbf{M}_\tau^{(1)*}(\mathbf{e}_C^1)^T$$

and

$$T'(s_2, l, C) = (\mathbb{O}, \mathbf{e}_{s_2})\mathbf{M}_\tau^{(32)*}(\mathbb{O}, \mathbf{e}_C^2)^T = \mathbf{e}_{s_2}\mathbf{M}_\tau^{(2)*}(\mathbf{e}_C^2)^T$$

are equal due to $(s_1, s_2) \in \mathcal{R}_{12}$.

For $l \neq \epsilon$,

$$T'(s_1, l, C) = (\mathbb{O}, \mathbf{e}_{s_1})\mathbf{M}_l^{(31)'}(\mathbb{O}, \mathbf{e}_C^1)^T = \mathbf{e}_{s_1}\mathbf{M}_l^{(1)'}(\mathbf{e}_C^1)^T$$

and

$$T'(s_2, l, C) = (\mathbb{O}, \mathbf{e}_{s_2})\mathbf{M}_l^{(32)'}(\mathbb{O}, \mathbf{e}_C^2)^T = \mathbf{e}_{s_2}\mathbf{M}_l^{(2)'}(\mathbf{e}_C^2)^T$$

are equal due to $(s_1, s_2) \in \mathcal{R}_{12}$.

Case 2: $s_1, s_2 \in \mathcal{S}_1 \cup \mathcal{S}_2$ and $C \subseteq \mathcal{S}_3$.
For $l = \epsilon$, $T'(s_1, l, C) = (\mathbb{O}, \mathbf{e}_{s_1})\mathbf{M}_\tau^{(31)*}(\mathbf{e}_C^3, \mathbb{O})^T = \mathbf{e}_{s_1}\mathbb{O}(\mathbf{e}_C^3)^T = \mathbb{O} = T'(s_2, l, C)$. The same holds for $l \neq \epsilon$, the reason is that matrices $\mathbf{M}_\tau^{(31)*}, \mathbf{M}_\tau^{(32)*}, \mathbf{M}_l^{(31)'}$, and $\mathbf{M}_l^{(32)'}$ all have a block of $\mathbb{O}$ entries for transitions from $\mathcal{S}_1 \cup \mathcal{S}_2$ to $\mathcal{S}_3$.

Case 3: $s_1, s_2 \in \mathcal{S}_3$ and $C \subseteq \mathcal{S}_1 \cup \mathcal{S}_2$.
Again, we need to consider $l = \epsilon$ and $l \neq \epsilon$. We start with the latter and distinguish whether the l-transition takes place in $\mathcal{A}_3$ or in $\mathcal{A}_1$, resp. $\mathcal{A}_2$. $s_1 = s_2$ holds due to $(s_1, s_2) \in \mathcal{R}_{123}$. We start with l in $\mathcal{A}_3$. Let $\mathbf{x}^{(1)} = \mathbf{M}_\tau^{(1)*} \hat{\cdot} (\mathbf{e}_C^{(1)})^T$ and $\mathbf{x}^{(2)} = \mathbf{M}_\tau^{(2)*} \hat{\cdot} (\mathbf{e}_C^{(2)})^T$, then $\mathcal{R}_{12}$ ensures that $\mathbf{x}^{(1)}(s) = \mathbf{x}^{(2)}(s')$ for any $(s, s') \in \mathcal{R}_{12}$ due to $T'(s, \epsilon, C) = T'(s', \epsilon, C)$. This is helpful to show

$$\begin{aligned}
T'(s_1, l, C) &= (\mathbf{e}_{s_1}, \mathbb{O})\mathbf{M}_l^{(31)'}(\mathbb{O}, \mathbf{e}_C^1)^T \\
&= \mathbf{e}_{s_1} \hat{\cdot} \mathbf{M}_l^{(3)'} \hat{\cdot} \mathbf{b}^{(3)} \hat{\cdot} \mathbf{a}^{(1)}\mathbf{M}_\tau^{(1)*} \hat{\cdot} (\mathbf{e}_C^1)^T \\
&= \mathbf{e}_{s_1} \hat{\cdot} \mathbf{M}_l^{(3)'} \hat{\cdot} \mathbf{b}^{(3)} \hat{\cdot} \widehat{\sum_{C' \in \mathcal{S}/\mathcal{R}_{12}}} \widehat{\sum_{s \in C'}} \mathbf{a}^{(1)}(s)\mathbf{x}^{(1)}(s) \\
&= \mathbf{e}_{s_1} \hat{\cdot} \mathbf{M}_l^{(3)'} \hat{\cdot} \mathbf{b}^{(3)} \hat{\cdot} \widehat{\sum_{C' \in \mathcal{S}/\mathcal{R}_{12}}} \widehat{\sum_{s \in C'}} \mathbf{a}^{(2)}(s)\mathbf{x}^{(2)}(s) \\
&= \mathbf{e}_{s_1} \hat{\cdot} \mathbf{M}_l^{(3)'} \hat{\cdot} \mathbf{b}^{(3)} \hat{\cdot} \mathbf{a}^{(2)}\mathbf{M}_\tau^{(2)*} \hat{\cdot} (\mathbf{e}_C^2)^T \\
&= \mathbf{e}_{s_2} \hat{\cdot} \mathbf{M}_l^{(3)'} \hat{\cdot} \mathbf{b}^{(3)} \hat{\cdot} \mathbf{a}^{(2)}\mathbf{M}_\tau^{(2)*} \hat{\cdot} (\mathbf{e}_C^2)^T \\
&= T'(s_2, l, C).
\end{aligned}$$

If the l-transition takes place in the second automaton we have an analogous situation. Now, let $\mathbf{x}^{(1)} = \mathbf{M}_l^{(1)'} \hat{\cdot} (\mathbf{e}_C^1)^T$ and $\mathbf{x}^{(2)} = \mathbf{M}_l^{(2)'} \hat{\cdot} (\mathbf{e}_C^2)^T$, then $\mathcal{R}_{12}$ ensures that

$\mathbf{x}^{(1)}(s) = \mathbf{x}^{(2)}(s')$ for any $(s, s') \in \mathcal{R}_{12}$ due to $T'(s, l, C) = T'(s', l, C)$.

$$\begin{aligned}
T'(s_1, l, C) &= (\mathbf{e}_{s_1}, \mathfrak{O})\mathbf{M}_l^{(31)'}(\mathfrak{O}, \mathbf{e}_C^1)^T \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{a}^{(1)}\mathbf{M}_l^{(1)'} \mathbin{\widehat{\cdot}} (\mathbf{e}_C^1)^T \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(3)} \mathbin{\widehat{\cdot}} \widehat{\sum_{C' \in \mathcal{S}/\mathcal{R}_{12}}} \widehat{\sum_{s \in C'}} \mathbf{a}^{(1)}(s)\mathbf{x}^{(1)}(s) \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(3)} \mathbin{\widehat{\cdot}} \widehat{\sum_{C' \in \mathcal{S}/\mathcal{R}_{12}}} \widehat{\sum_{s \in C'}} \mathbf{a}^{(2)}(s)\mathbf{x}^{(2)}(s) \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{a}^{(2)}\mathbf{M}_l^{(2)'} \mathbin{\widehat{\cdot}} (\mathbf{e}_C^2)^T \\
&= \mathbf{e}_{s_2} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{a}^{(2)}\mathbf{M}_l^{(2)'} \mathbin{\widehat{\cdot}} (\mathbf{e}_C^2)^T \\
&= T'(s_2, l, C).
\end{aligned}$$

It remains to consider $l = \epsilon$ for this case. We use again the observation for $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}$ with $\mathbf{x}^{(1)} = \mathbf{M}_\tau^{(1)*} \mathbin{\widehat{\cdot}} (\mathbf{e}_C^{(1)})^T$, $\mathbf{x}^{(2)}$ analogously. The argumentation only differs in the different matrices that result from using $\mathbf{M}_\tau^{(3i)*}$ instead of $\mathbf{M}_l^{(3i)'}$, $i = 1, 2$.

$$\begin{aligned}
T'(s_1, \epsilon, C) &= (\mathbf{e}_{s_1}, \mathfrak{O})\mathbf{M}_\tau^{(31)*}(\mathfrak{O}, \mathbf{e}_C^{(1)})^T \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{a}^{(1)}\mathbf{M}_\tau^{(1)*} \mathbin{\widehat{\cdot}} (\mathbf{e}_C^{(1)})^T \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(3)} \mathbin{\widehat{\cdot}} \widehat{\sum_{C' \in \mathcal{S}/\mathcal{R}_{12}}} \widehat{\sum_{s \in C'}} \mathbf{a}^{(1)}(s)\mathbf{x}^{(1)}(s) \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(3)} \mathbin{\widehat{\cdot}} \widehat{\sum_{C' \in \mathcal{S}/\mathcal{R}_{12}}} \widehat{\sum_{s \in C'}} \mathbf{a}^{(2)}(s)\mathbf{x}^{(2)}(s) \\
&= \mathbf{e}_{s_1} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{a}^{(2)}\mathbf{M}_\tau^{(2)*} \mathbin{\widehat{\cdot}} (\mathbf{e}_C^{(2)})^T \\
&= \mathbf{e}_{s_2} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} \mathbf{b}^{(3)} \mathbin{\widehat{\cdot}} \mathbf{a}^{(2)}\mathbf{M}_\tau^{(2)*} \mathbin{\widehat{\cdot}} (\mathbf{e}_C^{(2)})^T \\
&= T'(s_2, \epsilon, C).
\end{aligned}$$

Case 4: $s_1, s_2 \in \mathcal{S}_3$ and $C \subseteq \mathcal{S}_3$.
Considering matrices $\mathbf{M}_\tau^{(3i)*}$ and $\mathbf{M}_l^{(3i)'}$ for $i = 1, 2$, it is immediate that $T'(s_1, \epsilon, C) = T'(s_2, \epsilon, C)$ since $s_1 = s_2$ due to $\mathcal{R}_{123}$ and $\mathbf{M}_\tau^{(31)*}$ and $\mathbf{M}_\tau^{(32)*}$ contain the same block $\mathbf{M}_\tau^{(3)*}$ for $\mathcal{S}_3$, matrices $\mathbf{M}_l^{(31)'}$ and $\mathbf{M}_l^{(32)'}$ contain the same block $\mathbf{M}_l^{(3)'}$ for $\mathcal{S}_3$.

We finally have to prove that relation $\approx$ holds between the composed automata, i. e., that the condition of equal sums of initial weights according to the second part of Definition 3.1 are observed. It remains to show that $\mathcal{R}_{123}$ fulfills $\widehat{\sum}_{s \in C \cap (\mathcal{S}_1 \cup \mathcal{S}_3)} \alpha(s) = \widehat{\sum}_{s \in C \cap (\mathcal{S}_2 \cup \mathcal{S}_3)} \alpha(s)$ for all $C \in \mathcal{S}_{123}/\mathcal{R}_{123}$. The definition of initial weights assures that states receive initial weights that are either zero or are identical to the initial weights in the automaton before composition. If $\mathcal{A}_3$ is the first automaton in the composition, then only states from $\mathcal{S}_3$ have initial weights which are not zero. Since each class $C \cap \mathcal{S}_3$ equals $\{s_3\}$ or $\emptyset$, the condition holds. If $\mathcal{A}_1$ and $\mathcal{A}_2$ are the first

automata in the composition, then only states from $\mathcal{S}_1 \cup \mathcal{S}_2$ have non-zero initial weights. We have

$$\widehat{\sum_{s\in C\cap(\mathcal{S}_1\cup\mathcal{S}_3)}} \alpha(s) \;=\; \widehat{\sum_{s\in C\cap\mathcal{S}_1}} \alpha(s) \;=\; \widehat{\sum_{s\in C\cap\mathcal{S}_2}} \alpha(s) \;=\; \widehat{\sum_{s\in C\cap(\mathcal{S}_2\cup\mathcal{S}_3)}} \alpha(s). \qquad \square$$

The previous theorem shows that weak bisimulation is a congruence for direct sums and products of automata for commutative and idempotent semirings. The next theorem proves the same for the synchronized product of $\mathbb{K}i$-automata.

Theorem 4.2 *If $\mathcal{A}_1 \approx \mathcal{A}_2$ and $\mathcal{A}_3$ are finite $\mathbb{K}i$-automata, then*

$$\mathcal{A}_1||_{\mathcal{L}_c}\mathcal{A}_3 \approx \mathcal{A}_2||_{\mathcal{L}_c}\mathcal{A}_3 \quad \textit{and} \quad \mathcal{A}_3||_{\mathcal{L}_c}\mathcal{A}_1 \approx \mathcal{A}_3||_{\mathcal{L}_c}\mathcal{A}_4 \ .$$

Proof. (synchronized product)
Define relation by $\mathcal{R}_{123} \subseteq S_{123} \times S_{123}$ with $S_{123} = (S_1 \times S_3) \cup (S_2 \times S_3)$ and $((s_1, s_2), (s'_1, s'_2)) \in \mathcal{R}_{123}$ iff $(s_1, s'_1) \in \mathcal{R}_{12}$ and $s_2 = s'_2$. We prove that $\mathcal{R}_{123}$ is a weak bisimulation for $\mathcal{A}_1||_{\mathcal{L}_c}\mathcal{A}_3$ and $\mathcal{A}_2||_{\mathcal{L}_c}\mathcal{A}_3$ and $\mathcal{A}_1||_{\mathcal{L}_c}\mathcal{A}_3 \approx \mathcal{A}_2||_{\mathcal{L}_c}\mathcal{A}_3$, more precisely, we need to show that

1. $\mathcal{R}_{123}$ is an equivalence relation: it is reflexive, symmetric, transitive. This follows directly from $\mathcal{R}_{12}$ being an equivalence relation and using the identity relation for $\mathcal{A}_3$.

2. $\mathcal{R}_{123}$ is a weak bisimulation, i. e., $\forall (s, s') \in \mathcal{R}_{123} : (\alpha(s) = \alpha(s')) \wedge (\beta'(s) = \beta'(s')) \wedge T'(s_1, l, C) = T'(s_2, l, C))$ for all $C \in \mathcal{S}_{123}/\mathcal{R}_{123}$. Condition $\forall (s = (s_1, s_3), s' = (s_2, s_3)) \in \mathcal{R}_{123} : (\alpha(s) = \alpha(s'))$ follows directly from the definition of $\mathcal{R}_{123}$ using $\mathcal{R}_{12}$ and identity and the definition of α in the synchronized product, since for $(s_1, s_2) \in \mathcal{R}_{12}$ holds $\alpha(s_1) = \alpha(s_2)$, so $\alpha(s_1) \mathbin{\widehat{\cdot}} \alpha(s_3) = \alpha(s_2) \mathbin{\widehat{\cdot}} \alpha(s_3)$.

To prove $\beta'(s_1, s_3) = \beta'(s_2, s_3)$ with $(s_1, s_2) \in \mathcal{R}_{12}$ we have to consider the computation of the final weights. Due to commutativity and idempotency of the semiring we can apply Lemma 2.1 and matrix $\mathbf{M}_\tau^{(13)*}$ of the automaton composed of $\mathcal{A}_1$ and $\mathcal{A}_3$ can be represented as

$$\begin{aligned}
\mathbf{M}_\tau^{(13)*} &= \widehat{\sum_{k=0}^{\infty}} \left(\mathbf{M}_\tau^{(1)} \mathbin{\widehat{\otimes}} \mathbf{I} \mathbin{\widehat{+}} \mathbf{I} \mathbin{\widehat{\otimes}} \mathbf{M}_\tau^{(3)}\right)^k \\
&= \widehat{\sum_{k=0}^{\infty}} \widehat{\sum_{l=0}^{k}} \left(\mathbf{M}_\tau^{(1)}\right)^l \mathbin{\widehat{\otimes}} \left(\mathbf{M}_\tau^{(3)}\right)^{k-l} \\
&= \widehat{\sum_{k=0}^{\infty}} \left(\mathbf{M}_\tau^{(1)}\right)^k \mathbin{\widehat{\otimes}} \widehat{\sum_{l=0}^{\infty}} \left(\mathbf{M}_\tau^{(3)}\right)^l = \mathbf{M}_\tau^{(1)*} \mathbin{\widehat{\otimes}} \mathbf{M}_\tau^{(3)*}.
\end{aligned}$$

Similarly the matrix $\mathbf{M}_\tau^{(23)*}$ of the automaton composed of $\mathcal{A}_2$ and $\mathcal{A}_3$ equals $\mathbf{M}_\tau^{(2)*} \mathbin{\widehat{\otimes}} \mathbf{M}_\tau^{(3)*}$. For some state (s_i, s_3) with $s_i \in \mathcal{S}_i$, $i = 1, 2$, $\beta'(s_i, s_3)$ is given by

$$\begin{aligned}
&\mathbf{e}_{s_i} \mathbin{\widehat{\otimes}} \mathbf{e}_{s_3} \mathbin{\widehat{\cdot}} \left(\mathbf{M}_\tau^{(i)*} \mathbin{\widehat{\otimes}} \mathbf{M}_\tau^{(3)*}\right) \mathbin{\widehat{\cdot}} \mathbf{b}^{(i)} \mathbin{\widehat{\otimes}} \mathbf{b}^{(3)} \\
&\quad = \mathbf{e}_{s_i} \mathbf{M}_\tau^{(i)*} \mathbf{b}^{(i)} \mathbin{\widehat{\otimes}} \mathbf{e}_{s_3} \mathbf{M}_\tau^{(3)*} \mathbf{b}^{(3)} = \beta'_i(s_i) \mathbin{\widehat{\cdot}} \beta'_3(s_3).
\end{aligned}$$

The above identity holds since the two vector matrix vector products describe a scalar and the Kronecker product of two scalars equals the ordinary product. Since $\beta_i'(s_i)$ is identical for all weakly bisimilar states, equal final weights for states from an equivalence class of $C \in \mathcal{S}_{123}/\mathcal{R}_{123}$ are given.

Let us consider condition $T'(s,l,C) = T'(s',l,C)$. To prove the result, we use the matrix identities described above, in particular the representation of $\mathbf{M}_\tau^{(0)*} = \mathbf{M}_\tau^{(i)*} \mathbin{\widehat{\otimes}} \mathbf{M}_\tau^{(3)*}$ is relevant. All states of a $C \in \mathcal{S}_{123}/\mathcal{R}_{123}$ are of the form (s_i', s_3') for some fixed state s_3' and states s_i' are in a class $D \in \mathcal{S}_{12}/\mathcal{R}_{12}$.

Now assume that $l \in \mathcal{L}_c$, then each relevant path consists of a sequence of τ labelled transitions in both automata, followed by one synchronized l-labelled transitions. The corresponding weights can be expressed as

$$\begin{aligned}
&T(s,l,C)\\
&= (\mathbf{e}_{s_i} \mathbin{\widehat{\otimes}} \mathbf{e}_{s_3}) \mathbin{\widehat{\cdot}} \left(\mathbf{M}_\tau^{(i)*} \mathbin{\widehat{\otimes}} \mathbf{M}_\tau^{(3)*}\right) \mathbin{\widehat{\cdot}} \left(\mathbf{M}_l^{(i)} \mathbin{\widehat{\otimes}} \mathbf{M}_l^{(3)}\right) \mathbin{\widehat{\cdot}} \left(\mathbf{M}_\tau^{(i)*} \mathbin{\widehat{\otimes}} \mathbf{M}_\tau^{(3)*}\right)\\
&\quad \mathbin{\widehat{\cdot}} \left((\mathbf{e}_C)^T \mathbin{\widehat{\otimes}} (\mathbf{e}_{s_3'})^T\right)\\
&= \left(\mathbf{e}_{s_i} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(i)*} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(i)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(i)*} (\mathbf{e}_C)^T\right) \mathbin{\widehat{\otimes}} \left(\mathbf{e}_{s_3} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} \mathbf{M}_l^{(3)} \mathbin{\widehat{\cdot}} \mathbf{M}_\tau^{(3)*} \mathbin{\widehat{\cdot}} (\mathbf{e}_{s_3'})^T\right)\\
&= T'(s_1,l,C) \mathbin{\widehat{\cdot}} T'(s_3,l,s_3').
\end{aligned}$$

The first value is identical for all states from equivalence class D and the second part is unique for each equivalence class of $\mathcal{R}_{123}$.

For $l \notin \mathcal{L}_c$ we have to substitute matrix $\mathbf{M}_l^{(i)}$ or matrix $\mathbf{M}_l^{(3)}$ by an identity matrix. For $l = \epsilon$ $\mathbf{M}_\epsilon' = \mathbf{M}_\tau^{(i3)*} = \mathbf{M}_\tau^{(i)*} \mathbin{\widehat{\otimes}} \mathbf{M}_\tau^{(3)*}$ as shown above. Since $\mathcal{R}_{12}$ is a weak bisimulation (for $i = 1,2$) and D consists of a single state (for $i = 3$), the values $\mathbf{e}_{s_i} \mathbf{M}_\tau^{(i)*} (\mathbf{e}_C)^T = T'(s_i,\epsilon,C)$ are identical for weakly bisimilar states s_i. Consequently, also the products $T'(s_1,l,C) \mathbin{\widehat{\cdot}} T'(s_3,\epsilon,s_3')$, $T'(s_1,\epsilon,C) \mathbin{\widehat{\cdot}} T'(s_3,l,s_3')$ and $T'(s_1,\epsilon,C) \mathbin{\widehat{\cdot}} T'(s_3,\epsilon,s_3')$ are identical for all states from an equivalence class.

3. It remains to show that $\mathcal{R}_{123}$ fulfills $\widehat{\sum}_{s \in C \cap (\mathcal{S}_1 \cup \mathcal{S}_3)} \alpha(s) = \widehat{\sum}_{s \in C \cap (\mathcal{S}_2 \cup \mathcal{S}_3)} \alpha(s)$ for a class $C \in \mathcal{S}_{123}/\mathcal{R}_{123}$. By definition of $\mathcal{R}_{123}$, we obtain $\widehat{\sum}_{s \in C \cap (\mathcal{S}_1 \cup \mathcal{S}_3)} \alpha(s) = \widehat{\sum}_{(s_1,s_3) \in C \cap (\mathcal{S}_1 \cup \mathcal{S}_3)} \alpha(s_1) \mathbin{\widehat{\cdot}} \alpha(s_3)$ and analogously for $(s_2,s_3) \in C$. However all $(s_1,s_3),(s_2,s_3) \in C$ vary only by s_1, s_2, since s_3 is fixed per equivalence class. Hence $\alpha(C_1) = \alpha(C_2)$ implies the required property.

The proof for the second part, namely the synchronized product with $\mathcal{A}_3$ in the first position is completely analogous, because the Kronecker sum observes some form of pseudo-commutativity which means that $\mathbf{A} \mathbin{\widehat{\otimes}} \mathbf{B}$ and $\mathbf{B} \mathbin{\widehat{\otimes}} \mathbf{A}$ are identical up to a symmetric permutation of rows and columns [15]. □

The proofs of the above theorem are based on the idempotency of addition in the considered semiring. So the theorem holds for semirings like $\max/+$. For non-idempotent semirings, as it is the case for probabilistic or stochastic automata, we do not provide results.

5. Equivalence under Choice

It might be surprising that our notion of weak bisimulation is a congruence whereas weak bisimulation for the boolean semiring is not a congruence for the process algebra CCS. The congruence property of weak bisimulation for CCS is destroyed by the choice operator, for all other operators it is a congruence. This is the motivation for defining weak congruence in [21] that achieves a congruence relation for CCS. If weights are introduced into a process algebra like for instance in GPA [8], again the choice operator causes similar problems as in Milner's approach.

However, for the composition of automata, the choice operator is not common and it depends on its definition whether weak bisimulation is a congruence or not. In this section, we define a choice operator that appears natural for automata and that also makes weak bisimulation a congruence. If two automata $\mathcal{A}_1, \mathcal{A}_2$ are composed by choice, denoted by $\vee$ (not $+$ to avoid confusion with direct sum of automata), we believe that the resulting automaton should make an either-or decision whether the future behaviour should be that of $\mathcal{A}_1$ or $\mathcal{A}_2$. Hence, the composed automaton $\mathcal{A}_1 \vee \mathcal{A}_2$ starts in a state where it can either perform transitions of the initial state of $\mathcal{A}_1$ or $\mathcal{A}_2$. This form of choice differs from the choice operator in CCS because it combines initial states, whereas the choice operator of CCS assures that the initial state of the composed system is left after the first transition and the composed system behaves like one of the systems that have been composed. To notice the difference, one can consider the composition of two automata with a single state and one transition labelled with a and b, respectively. Our definition of choice would result in a composed automaton with one state and one a- and one b-labelled transition. In CCS, the resulting system would have three states, the initial state with outgoing a- and b-labelled transitions that ends in a state with one cyclic transition labelled with a or b, respectively.

Definition 5.1 *Let $\mathcal{A}_1, \mathcal{A}_2$ be $\mathbb{K}$-automata with unique initial states s_1^0, s_2^0 with $\alpha(s_1^0) = \alpha(s_2^0) = \mathbb{1}$, $\beta(s_1^0) = \beta(s_2^0)$ and, for all other states $s_i \in \mathcal{S}_1 \cup \mathcal{S}_2 \setminus \{s_1^0, s_2^0\}$, $\alpha(s_i) = \mathbb{0}$.*

$\mathcal{A}_0 = \mathcal{A}_1 \vee \mathcal{A}_2$ results from $\mathcal{A}_1$ and $\mathcal{A}_2$ by gluing together the initial states s_1^0 and s_2^0 to a new state s_0^0 such that all transitions leaving s_i^0 in $\mathcal{A}_i$ leave s_0^0 in $\mathcal{A}_0$ and all transitions entering s_i^0 in $\mathcal{A}_i$ enter s_0^0 in $\mathcal{A}_0$. All remaining transitions remain unchanged. $\alpha(s_0^0) = \mathbb{1}$ and $\beta(s_0^0) = \beta(s_1^0)$.

Consequently, for the state space the relation $\mathcal{S}_0 = \{s_0^0\} \cup \mathcal{S}_1 \cup \mathcal{S}_2 \setminus \{s_1^0, s_2^0\}$ holds.

Weak bisimulation is also a congruence for the choice operator that we defined for automata. Ahead of any formal considerations, we consider the classical example in concurrency theory that illustrates the compositionality defect of choice in the context of a process algebra like CCS. Let $\mathcal{A}_1$ be an automaton that only performs a single transition with label a and then stops. Let $\mathcal{A}_2$ be an automaton that only performs a sequence of a single τ transition followed by a transition a and then stops. Finally, let $\mathcal{A}_3$ be an automaton that only performs a single transition b and then stops. According to the definition of weak bisimulation $\approx_{CCS}$ in CCS, $\mathcal{A}_1 \approx_{CCS} \mathcal{A}_2$ but a composition with $\mathcal{A}_3$ by choice, yields automata that are not weakly bisimilar. In our

case, $\mathcal{A}_1 \not\approx \mathcal{A}_2$ from the beginning if $\mathcal{A}_1$, $\mathcal{A}_2$ fulfill the restrictions of choice. $\mathcal{A}_1 \not\approx \mathcal{A}_2$ then holds because the unique initial states are the only ones with $\alpha(s_1^0) = \alpha(s_2^0) = \mathbb{1}$ and the state reached by the τ transition in $\mathcal{A}_2$ is not weakly bisimilar to any state in $\mathcal{A}_1$. So in our case, the definition of choice prevents that known difficulty.

The following theorem states the congruence property with respect to choice.

Theorem 5.1 *Let $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$ with unique initial states such the choice operation $\vee$ is defined and $\mathcal{A}_1 \approx \mathcal{A}_2$, then $\mathcal{A}_1 \vee \mathcal{A}_3 \approx \mathcal{A}_2 \vee \mathcal{A}_3$.*

Proof. Let $\mathcal{A}_{13} = \mathcal{A}_1 \vee \mathcal{A}_3$ and $\mathcal{A}_{23} = \mathcal{A}_2 \vee \mathcal{A}_3$. Definition of $\vee$ implies that initial states are unique and $\alpha(s_1^0) = \alpha(s_2^0) = \alpha(s_3^0) = \alpha(s_{13}^0) = \alpha(s_{23}^0) = \mathbb{1}$ and $\mathbb{0}$ for all other states of $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_{13}, \mathcal{A}_{23}$. Consequently, for any weak bisimulation, the initial states must be the only elements in their classes, i. e., $\mathcal{R}_{12}(s_1^0, s_2^0)$ are the only elements in their class. Since only initial states are merged, it is sufficient to prove that choice does not destroy $\mathcal{R}_{1323}(s_{13}^0, s_{23}^0)$ to show that $\mathcal{R}_{1323} = \mathcal{R}_{12} \cup^* \mathrm{id}_3$ is a bisimulation. We need to prove the 3 conditions according to the definition of weak bisimulation:

1) By definition, $\alpha(s_{13}^0) = \alpha(s_{23}^0) = \mathbb{1}$.

2) We need to prove that $\beta'(s_{13}^0) = \beta'(s_{23}^0)$. We assume that $T'_{13}(s_{13}^0, \epsilon, s_{13}^0) = T'_{23}(s_{23}^0, \epsilon, s_{23}^0)$. With this assumption, it is straightforward that

$$\begin{aligned}
\beta'(s_{13}^0) &= \widehat{\sum_{s \in \mathcal{S}_{13}}} T'_{13}(s_{13}^0, \epsilon, s_{13}^0) \mathbin{\hat{\cdot}} T'_{13}(s_{13}^0, \epsilon, s) \mathbin{\hat{\cdot}} \beta(s) \\
&= \widehat{\sum_{s \in \mathcal{S}_1}} T'_{13}(s_{13}^0, \epsilon, s_{13}^0) \mathbin{\hat{\cdot}} T'_1(s_1^0, \epsilon, s) \mathbin{\hat{\cdot}} \beta(s) \\
&\quad \mathbin{\hat{+}} \widehat{\sum_{s \in \mathcal{S}_3}} T'_{13}(s_{13}^0, \epsilon, s_{13}^0) \mathbin{\hat{\cdot}} T'_3(s_3^0, \epsilon, s) \mathbin{\hat{\cdot}} \beta(s) \\
&= T'_{13}(s_{13}^0, \epsilon, s_{13}^0) \mathbin{\hat{\cdot}} (\beta'(s_1^0) \mathbin{\hat{+}} \beta'(s_3^0)) \\
&= T'_{13}(s_{13}^0, \epsilon, s_{13}^0) \mathbin{\hat{\cdot}} (\beta'(s_2^0) \mathbin{\hat{+}} \beta'(s_3^0)) \\
&= T'_{13}(s_{23}^0, \epsilon, s_{23}^0) \mathbin{\hat{\cdot}} (\beta'(s_2^0) \mathbin{\hat{+}} \beta'(s_3^0)) = \beta'(s_{23}^0).
\end{aligned}$$

To prove $T'_{13}(s_{13}^0, \epsilon, s_{13}^0) = T'_{23}(s_{23}^0, \epsilon, s_{23}^0)$ we have to consider the possible paths from s_{i3}^0 to s_{i3}^0 via τ transitions only. Each path consists of k subpaths which visit states in one of the automata only and each of these subpaths can visit arbitrarily often the initial state. By help of Lemma 2.1, $T'_i(s_i^0, \epsilon, s_i^0) = (T'_i(s_i^0, \epsilon, s_i^0))^2$ and idempotency of addition, the weights of the paths can be computed as follows.

$$\begin{aligned}
T'_{i3}(s_{i3}^0, \epsilon, s_{i3}^0) &= \widehat{\sum}_{k=0}^{\infty} \left(T'_i(s_i^0, \epsilon, s_i^0) \mathbin{\hat{\cdot}} T'_3(s_3^0, \epsilon, s_3^0)\right)^k \\
&= \widehat{\sum}_{k=0}^{\infty} \left(T'_i(s_i^0, \epsilon, s_i^0)\right)^k \mathbin{\hat{\cdot}} \widehat{\sum}_{l=0}^{\infty} \left(T'_3(s_3^0, \epsilon, s_3^0)\right)^l \\
&= T'_i(s_i^0, \epsilon, s_i^0) \mathbin{\hat{\cdot}} T'_3(s_3^0, \epsilon, s_3^0).
\end{aligned}$$

Since $T'_1(s_1^0, \epsilon, s_1^0) = T'_2(s_2^0, \epsilon, s_2^0)$ by $\mathcal{A}_1 \approx \mathcal{A}_2$, the assumption holds.

3) It remains to prove that $T'_{13}(s^0_{13}, l, C) = T'_{23}(s^0_{23}, l, C)$.

$$\begin{aligned} T'_{13}(s^0_{13}, l, C) &= T'_{13}(s^0_{13}, \epsilon, s^0_{13}) \mathbin{\hat{\cdot}} T'_1(s^0_{13}, l, C_1) \mathbin{\hat{+}} T'_{13}(s^0_{13}, \epsilon, s^0_{13}) \mathbin{\hat{\cdot}} T'_3(s^0_{13}, l, C_3) \\ &= T'_{13}(s^0_{13}, \epsilon, s^0_{13}) \mathbin{\hat{\cdot}} T'_2(s^0_{23}, l, C_2) \mathbin{\hat{+}} T'_{13}(s^0_{13}, \epsilon, s^0_{13}) \mathbin{\hat{\cdot}} T'_3(s^0_{23}, l, C_3) \\ &= T'_{23}(s^0_{23}, \epsilon, s^0_{23}) \mathbin{\hat{\cdot}} T'_2(s^0_{23}, l, C_2) \mathbin{\hat{+}} T'_{13}(s^0_{23}, \epsilon, s^0_{23}) \mathbin{\hat{\cdot}} T'_3(s^0_{23}, l, C_3) \\ &= T'_{23}(s^0_{23}, l, C), \end{aligned}$$

since we have $T'_1(s^0_{13}, l, C_1) = T'_2(s^0_{23}, l, C_2)$ according to $\mathcal{A}_1 \approx \mathcal{A}_2$, $T'_3(s^0_{13}, l, C_3) = T'_3(s^0_{23}, l, C_3)$ by construction, and $T'_{13}(s^0_{13}, \epsilon, s^0_{13}) = T'_{23}(s^0_{23}, \epsilon, s^0_{23})$ according to the argumentation for $\beta'(s^0_{13}) = \beta'(s^0_{23})$ above. □

6. Examples

In this section, we consider different examples for $\mathbb{K}i$-automata to show the wide applicability of this class of models. Further applications of $\mathbb{K}i$-automata can be found in the textbook [3] which gives a comprehensive overview of the modelling of discrete event systems by means of max/+-systems and related models. These models are a subclass of our weighted automata model.

6.1. A Small Automaton with Different Interpretations

We begin with a very simple model to show how corresponding bisimilar automata differ if different semirings are used for the valuation of transitions. The example automaton is shown in Figure 1. We present the example in a graphical manner,

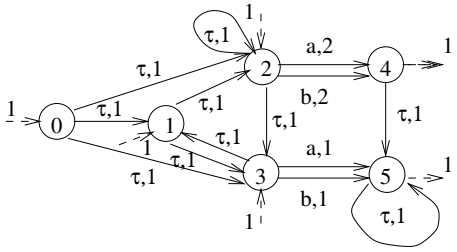

Figure 1: Example automaton $\mathcal{A}_1$

which is often easier to understand than a matrix representation. States are numbered consecutively 0 through $n-1$ and arcs between states describe transitions. Arcs are annotated with a symbol from $\mathcal{L}$ and a weight from $\mathbb{K}$. All connections that are not drawn are implicitly weighted with $\mathbb{0}$. For a state s, we visualize $\alpha(s) \neq \mathbb{0}$ by a dashed arc that points to s and that is annotated by $\alpha(s)$. An arc for $\alpha(s) = \mathbb{0}$ is omitted. Similarly, outgoing dashed arcs visualize $\beta(s)$, those arcs are annotated by $\beta(s) \neq \mathbb{0}$. For states without outgoing dashed arc the value of β equals $\mathbb{0}$. In our example, the alphabet includes symbols a, b, τ and weights are real numbers. We will analyze the example for different semirings to illustrate the differences. Matrix $\mathbf{M}^{(1)}_\tau$

and $\mathbf{M}_\tau^{(1)*}$ have the following structure

$$\mathbf{M}_\tau^{(1)} = \begin{pmatrix} \mathbb{0} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \mathbb{0} & \mathbb{1} & \mathbb{1} & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \mathbb{0} & \mathbb{1} & \mathbb{1} & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \mathbb{1} & \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{1} \\ \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{1} \end{pmatrix}, \quad \mathbf{M}_\tau^{(1)*} = \begin{pmatrix} \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{1} & \mathbb{1} \\ \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{0} & \mathbb{1} \end{pmatrix}$$

where the concrete values for $\mathbb{0}$ and $\mathbb{1}$ depend on the semiring.

First, we consider the automaton over the semiring $(\mathbb{R} \cup \{\pm\infty\}, \max, +, -\infty, 0)$. For this semiring, all arcs that are not drawn have weight $-\infty$. The largest bisimulation for this semiring includes the equivalence classes $\{\{0,1,2,3\},\{4,5\}\}$, a weakly equivalent automaton $\mathcal{A}_2$ with only two states is shown on the left side of Figure 2. For the semiring $(\mathbb{R} \cup \{\infty\}, \min, +, \infty, 0)$, the largest weak bisimulation contains 3

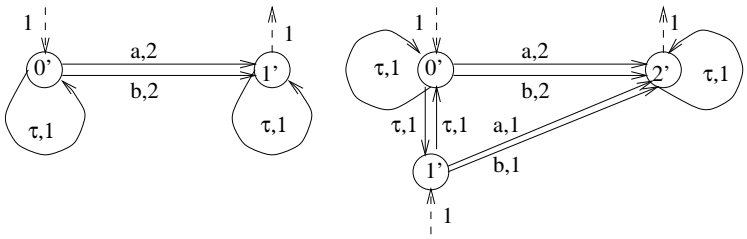

Figure 2: Weakly bisimilar automata $\mathcal{A}_2$ for max /+ and $\mathcal{A}_3$ for min /+

classes, namely $\{\{0,1,2\},\{3\},\{4,5\}\}$. A weakly bisimilar automaton $\mathcal{A}_3$ with 3 states is shown on the right side of Figure 2.

For the semirings $(\mathbb{R}\cup\{\pm\infty\}, \min, \max, \infty, -\infty)$ and $(\mathbb{R}\cup\{\pm\infty\}, \max, \min, -\infty, \infty)$ the largest bisimulations consist of the equivalence classes $\{\{0,1,2,3\},\{4,5\}\}$ and $\{\{2\},\{0,1,3\},\{4,5\}\}$, respectively. Minimal weakly bisimilar automata for these semirings are shown in Figure 3.

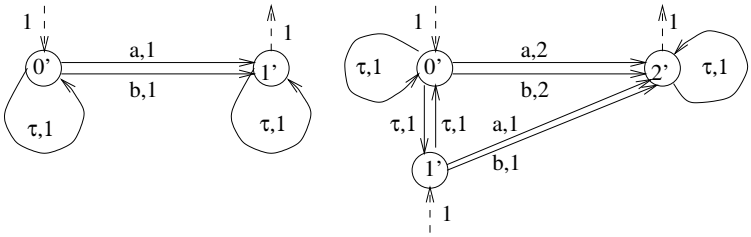

Figure 3: Weakly bisimilar automata $\mathcal{A}_4$ for min / max and $\mathcal{A}_5$ for max / min

6.2. Shortest Paths

For the computation of the shortest path from one city to another, $\mathbb{K}i$-automata over the semiring $(\mathbb{R} \cup \{\infty\}, \min, +, \infty, 0)$ are commonly applied [3]. The states of

an automaton correspond to cities and an arc with weight k between states i and j describes the length of the path between city i and j. For the computation of the shortest path between i and j, $\alpha(i) = \mathbb{1}$ (i. e., in the min/+ semiring $\mathbb{1} = 0$) and $\beta(j) = \mathbb{1}$, all remaining values for α and β are $\mathbb{0}$ (i. e., in the min/+ semiring $\mathbb{0} = \infty$). Without further constraints, all transitions of the automaton are labelled with τ and the length of the shortest path from i to j is given by

$$\mathbf{a}\mathbf{M}_\tau^*\mathbf{b} = \mathbf{a}\left(\widehat{\sum_{k=1}^{n-1}}(\mathbf{M}_\tau)^k\right)\mathbf{b}.$$

The latter equality holds since in a graph with n nodes and non-negative arc weights, the shortest path can not contain more than $n-1$ arcs and $\mathbf{a}(\mathbf{M}_\tau)^k\mathbf{b}$ describes the shortest path from i to j via $k-1$ intermediate cities. Since summation preserves the minimum, after at most $n-1$ iterations the result is the length of the shortest path. Using the results of Theorem 3.1, the length of the shortest path is preserved by weak bisimulation such that the automaton describing the connection can first be reduced and the shortest path can be computed afterwards. Since bisimulation is a congruence according to our composition operators, it can be applied in a compositional way. For shortest path problems, direct sum and product are common composition operations. The direct sum of two automata is usable for specifying alternative sets of routes, where first a decision among a set of routes has to be taken and then the shortest path in the set is found. The direct (or cascaded) product describes the concatenation of paths through a single node. The node through which all paths go, is used as only end node in the first automaton (i. e., the final weight is set to $\mathbb{1}$) and it is used as only initial node in the second automaton (i. e., the initial weight is set to $\mathbb{1}$). Afterwards, the automaton $\mathcal{A}_1 \mathbin{\widehat{\cdot}} \mathcal{A}_2$ is built to describe the concatenated paths.

Usually all paths are labelled with τ, since there are no specific constraints defined. However, one can think of extending the problem by using weights for durations and labels from some alphabet to represent means of transport. Label τ might be used to represent connections with means of transport that are not relevant for travelling decisions. A small example can be found in Figure 4. In this example apart from τ-labelled internal transitions, transitions with labels s (ship) and p (plane) exist. Automaton $\mathcal{A}_{\text{map}}$ describes the original plan with 5 cities. The relation $\{\{0\},\{1,2,3\},\{4\}\}$ is a weak bisimulation for this automaton and one possible weakly bisimilar automaton $\mathcal{A}_{\text{map}'}$ is shown. Travelling decisions can then be made by defining schedulers which are composed via the synchronized product with the automaton describing the paths. For the small example two different schedulers are presented in Figure 4. The first scheduler allows for an arbitrary use of ship or plane, whereas the second scheduler allows the use of one plane and one ship. In both cases, the scheduler $\mathcal{A}_{\text{sched}}$ is composed with the automaton $\mathcal{A}_{\text{map}}$ describing the path length via $\mathcal{A}_{\text{map}} \| _{\{s,p\}} \mathcal{A}_{\text{sched}}$. Since weak bisimulation is a congruence for the parallel composition and minimal paths are preserved by weak bisimulation, both automata can be first reduced by weak bisimulation before composition.

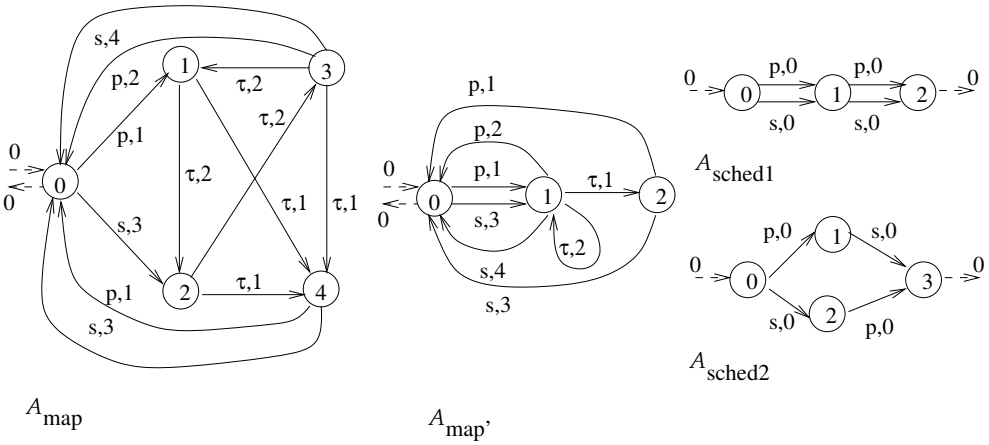

Figure 4: Automaton $\mathcal{A}_{\text{map}}$ and weakly bisimilar automaton $\mathcal{A}_{\text{map}'}$ and two possible scheduler for min /+

6.3. Performance Evaluation of Discrete Event Systems

Performance evaluation of discrete event systems using $(\max, +)$-automata is introduced in [13]. The class of models considered in that approach corresponds to our automata model over the semiring $(\mathbb{R} \cup \{\pm\infty\}, \max, +, -\infty, 0)$. This class of models combines two approaches that are commonly used for the analysis of discrete event systems, namely conventional automata for the description of the logical behaviour and max /+-systems for the quantitative analysis of so called timed event graphs [3]. The use of $(\max, +)$-automata allows one to describe very general models with nondeterminism and concurrency, which are made deterministic by an appropriate scheduler. The use of weak bisimulation is very helpful in this context since it defines naturally equivalent behaviour and it is usable in compositional modelling.

An example of a workshop with two machines that process three types of parts under two working regimes a and b are shown in Figure 5. At working regime a, a

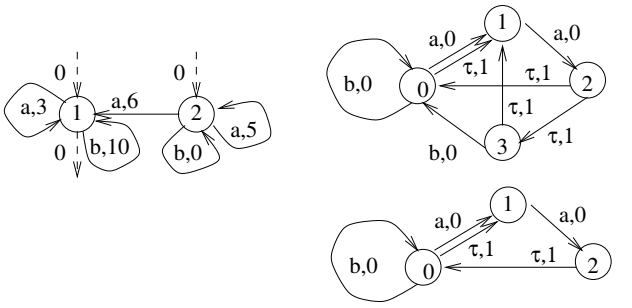

Figure 5: Workshop and two weakly bisimilar schedulers

part is processed on machine 2 during 5 units of time, then it is sent to machine 1

which takes 1 unit of time. Machine 1 processes first one part in 3 units of time and combines this part with the part sent by machine 2. Under working regime b machine 1 processes one part requiring 10 units of time and machine 2 is idle. The example is taken from [13]. The system is driven by some scheduler that generates sequences of a and b values specifying sequences of working regimes. In addition to [13], we also allow the scheduler to perform some τ action, which describes indeterminism or some external delays. In the example, both schedulers are weakly bisimilar and accept the language a^*b^*. However, observe that the language of an automaton with transition weights does not describe the behaviour completely, since weights have to be considered in addition. A scheduler is composed with the automaton by the synchronized product of automata that is described by the Kronecker product of the automata matrices. The fact that a Kronecker operation describes the composition is already noticed in [13]. The resulting automaton can afterwards be analyzed according to the worst case and best case behaviour under allowed schedules. For details about the corresponding analysis methods, we refer to [13] and the references therein.

6.4. Analysis of Timed Event Graphs

Timed event graphs are a class of timed Petri Nets with synchronization but without choices. For timed event graphs, $\max/+$-algebra can be used to determine performance measures like mean cycle times. Here, we only give a brief example how timed event graphs can be analyzed as linear $\max/+$-systems and how this analysis profits from the use of weak bisimulation. We refer to [3] for a comprehensive overview of the analysis of timed event graphs by methods from $\max/+$-algebra.

We assume in the sequel that the reader is familiar with basic terminology of Petri nets. The class of nets that we consider associates time with places, transitions fire instantaneously if they are enabled. Consequently, a token that arrives at a place is frozen there for some time and when this time is over, the token is available to be absorbed by transitions in the post-set of the place. Since the nets are timed event graphs there is no choice between transitions for a token at a place: each place has at most one transition in its post set. First, we assume that times are constant.

A simple example net is shown in Figure 6. For an analysis via $\max/+$, we asso-

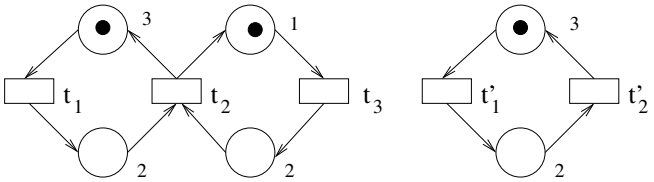

Figure 6: Two weakly bisimilar timed event graphs

ciate a variable with each transition and assign an index to each variable as for its corresponding transition. For the small system shown on the left side of Figure 6, we obtain 3 variables. Let $\mathbf{x}^{(k)}$ be the state vector that describes the times of the k-th firing of transitions. Thus, $\mathbf{x}^{(k)}(1)$ contains the time of the k-th firing of transition

t_1. It is easy to prove that the following recurrence relations between the firing times exist where $\widehat{\cdot}$ equals + and $\widehat{+}$ equals max

$$\begin{aligned}
\mathbf{x}^{(k)}(1) &= 3 \mathbin{\widehat{\cdot}} \mathbf{x}^{(k-1)}(2),\\
\mathbf{x}^{(k)}(2) &= 2 \mathbin{\widehat{\cdot}} \mathbf{x}^{(k)}(1) \mathbin{\widehat{+}} 2 \mathbin{\widehat{\cdot}} \mathbf{x}^{(k)}(3),\\
\mathbf{x}^{(k)}(3) &= 1 \mathbin{\widehat{\cdot}} \mathbf{x}^{(k-1)}(2).
\end{aligned}$$

It is interesting to note that only recurrences to the k-th and the $k-1$-th firing occur in the equations. We collect recurrences to k-th firing in matrix $\mathbf{M}_0$ and recurrences to $k-1$ in matrix $\mathbf{M}_1$. This yields the following matrices where $\mathbf{M}_i(u,v)$ denotes the time that a token is frozen after the $k-i$-th firing of transition u which is used for the k-th firing of transition v.

$$\mathbf{M}_0 = \begin{pmatrix} -\infty & 2 & -\infty \\ -\infty & -\infty & -\infty \\ -\infty & 2 & -\infty \end{pmatrix} \quad \text{and} \quad \mathbf{M}_1 = \begin{pmatrix} -\infty & -\infty & -\infty \\ 3 & -\infty & 1 \\ -\infty & -\infty & -\infty \end{pmatrix}.$$

The dynamics of the net is then given by the following relation (for details see [3, Chapter 2]).

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k)} \mathbin{\widehat{\cdot}} \mathbf{M}_0 \mathbin{\widehat{+}} \mathbf{x}^{(k-1)} \mathbin{\widehat{\cdot}} \mathbf{M}_1 = \mathbf{x}^{(k-1)} \mathbin{\widehat{\cdot}} \mathbf{M}_1 \mathbin{\widehat{\cdot}} \mathbf{M}_0^*$$

where $\mathbf{x}^{(0)}$ is the zero vector.

Although timed event systems do not contain labelled transitions, we can interpret the transitions in $\mathbf{M}_0$ as τ-labelled which means that we only observe transition t_2 explicitly. However, this observation is enough to determine the cycle time of the net. Matrix $\mathbf{M}_0^*$ has the structure

$$\mathbf{M}_0^* = \begin{pmatrix} 0 & 2 & -\infty \\ -\infty & 0 & -\infty \\ -\infty & 2 & 0 \end{pmatrix}$$

such that $\{\{t_1, t_3\}, \{t_2\}\}$ is a weak bisimulation and the net on the right side of Figure 6 is weakly bisimilar to the original net.

We presented the approach by means of a very simple example. It can, of course, be extended. Labelling of transitions can be made more explicit by labelling transitions of the net. Furthermore, we can consider more general classes of nets where conflicting transitions are labelled with different labels and conflicts are explicitly resolved by some scheduler similar to the example presented before. In this case, weak bisimulation is used to reduce the complexity of the net and the reduced model is afterwards composed with a scheduler via synchronized composition.

We have considered constant durations in our example. The theory also applies to stochastic holding times in the places described by random variables and FIFO ordering of tokens in a place. The definition of weak bisimulation can be extended to stochastic systems if the maximum of random variables $X_1, \ldots, X_K$ can be characterized by some random variable Y, i. e., $\max(X_1, \ldots, X_K) = Y$ which is possible if $Y = X_k$ for some $k \in \{1, \ldots, K\}$ like for constant durations. However, for general distributions the definition of Y is much more complex and often impossible.

7. Conclusion

In this paper, we introduced an extension of weak bisimulation to weighted automata over commutative and idempotent semirings. The weak bisimulation is shown to be a congruence for several composition operators defined for automata, namely direct sum, direct or cascaded product, synchronized product and a specific choice operator. For the boolean semiring, which is included in the class of semirings we consider, our definition of weak bisimulation corresponds to the usual definition of weak bisimulation for untimed automata or process algebras. It has been shown that the largest weak bisimulation can be defined as a fixed point of a partition refinement. Therefore an algorithm to compute the largest bisimulation is easy to realize.

There are several topics for future research. It would be interesting to consider weak bisimulation for more general semirings that are not necessarily idempotent. There is existing work [4] with a very similar definition of weak bisimulation for fully probabilistic processes, where the semiring is not idempotent. However, the interesting aspect for more general semirings is, which properties are preserved by weak bisimulation, in particular after composition of automata. As mentioned in [4], weak probabilistic bisimulation is not preserved by synchronized products (i. e., parallel composition in the context of probabilistic processes). The proofs we present in this paper often use properties that are specific to commutative and idempotent semirings and are therefore not valid for more general structures.

There exists a variety of analysis algorithms for $\max/+$-systems that are available and commonly used for analysis of those systems. These algorithms compute eigenvectors of matrices or solution of sets of linear or nonlinear equations in the $\max/+$ semiring (see [3] for details). By means of weak bisimulation, it is possible to reduce transition matrices for those systems while preserving their behaviour. It is an interesting issue for further research to investigate which results are preserved by weak bisimulation in the analysis of those systems.

Acknowledgements

We thank the two anonymous referees who gave in-depth reviews in extraordinary short time; their constructive comments are kindly acknowledged.

References

[1] S. Andova, J. C. M. Baeten, Abstraction in probabilistic process algebras. In: *Proc. Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*. LNCS **2031**, Springer-Verlag, 2001, 204–219.

[2] A. Arnold, *Finite Transition Systems*. Prentice Hall, 1994.

[3] F. Baccelli, G. Cohen, G. Olsder, J. Quadrat, *Synchronization and Linearity*. John Wiley and Sons, 1992.

[4] C. Baier, H. Hermanns, Weak bisimulation of fully probabilistic processes. In: *Proc. Computer Aided Verification* (*CAV*). LNCS **1254**, Springer-Verlag, 1997, 119–130.

[5] P. Buchholz, Markovian process algebra: composition and equivalence. In: U. Herzog, M. Rettelbach (eds.), Proc. of the 2nd Work. on Process Algebras and Performance Modelling. Arbeitsberichte des IMMD, University of Erlangen, no. 27, 1994, 11–30.

[6] P. Buchholz, Bisimulation for automata with transition costs. Submitted for publication, 2000.

[7] P. Buchholz, Efficient computation of equivalent and reduced representations for stochastic automata. *Int. Journ. Computer Systems Science and Engineering* **15** (2000) 2, 93–103.

[8] P. Buchholz, P. Kemper, Quantifying the dynamic behavior of process algebras. In: L. de Alfaro, S. Gilmore (eds), *Process Algebras and Probabilistic Methods*. LNCS **2165**, Springer-Verlag, 2001, 184–199.

[9] R. G. Bukharaev, *Theorie der stochastischen Automaten*. Teubner, 1995.

[10] R. Cleaveland, J. Parrow, B. Steffen, The concurrency workbench: a semantics based tool for the verification of concurrent systems. *ACM Transactions on Programming Languages and Systems* **15** (1993) 1, 36–72.

[11] S. Eilenberg, *Automata, Languages and Machines*. Academic Press, 1974.

[12] J. C. Fernandez, An implementation of an efficient algorithm for bisimulation equivalence. *Science of Computer Programming* **13** (1989/90), 219–236.

[13] S. Gaubert, Performance evaluation of (max/+) automata. *IEEE Transactions on Automatic Control* **40** (1995) 12, 2014–2025.

[14] S. Gaubert, Max Plus, Methods and applications of (max,+) linear algebra. Research Report **3088**, INRIA, 1997.

[15] A. Graham, *Kronecker products and matrix calculus with applications*. Ellis Howard, 1981.

[16] H. Hermanns, U. Herzog, V. Mertsiotakis, Stochastic process algebras – between LOTOS and Markov chains. *Computer Networks and ISDN Systems* **30** (1998) 9/10, 901–924.

[17] J. Hillston, A compositional approach for performance modelling. PhD Thesis, Dep. of Comp. Sc., University of Edinburgh, 1994.

[18] J. Karhumäki, W. Plandowski, W. Rytter, Pattern matching problems for 2-dimensional images described by finite automata. *Nordic Journal of Computing* **7** (2000).

[19] W. Kuich, A. Salomaa, *Semirings, Automata, Languages*. ETACS Monographs on Theoretical Computer Science, Springer-Verlag, 1986.

[20] K. Larsen, A. Skou, Bisimulation through probabilistic testing. *Information and Computation* **94** (1991), 1–28.

[21] R. MILNER, *Communication and concurrency*. Prentice Hall, 1989.

[22] M. MOHRI, Finite-state transducers in language and speech processing. *Computational Linguistics* **23** (1997) 2.

[23] M. MOHRI, F. C. N. PEREIRA, M. RILEY, The design principles of a weighted finite-state transducer library. *Theoretical Computer Science* **231** (2000), 17–32.

[24] R. PAIGE, R. E. TARJAN, Three partition refinement algorithms. *SIAM Journal on Computing* **16** (1987) 6, 973–989.

[25] D. PARK, Concurrency and automata on infinite sequences. In: *Proc. 5th GI Conference on Theoretical Computer Science*. LNCS **104**, Springer-Verlag, 1981, 167–183.

(Received: April 19, 2002; revised: March 4, 2003)