Synchronous composition of interval weighted automata

Jan Komenda * **Sébastien Lahaye** ** **Jean-Louis Boimond** **

* *Institute of Mathematics, Czech Academy of Sciences, Zizkova 22, 616 62 Brno, Czech Republic (e-mail: komenda@ipm.cz)*
** *LISA Angers, 62, Avenue Notre Dame du Lac, 49000 Angers, France (e-mail: { sebastien.lahaye, jean-louis.boimond}@istia.univ-angers.fr)*

Abstract: Interval weighted automata are introduced as automata with weights in a product dioid (idempotent semiring). They constitute an extension of (max,+) automata since they enable us to model temporal constraints (instead of exact durations) for transitions. Their synchronous composition, which coincides with the synchronous product of underlying (one clock) timed automata, results in multi-event interval weighted automata, whose behaviors are studied. Our approach is illustrated by an example.

Keywords: Timed automata, (max,+) automata, synchronous product

1. INTRODUCTION

In the theory of timed discrete-event systems the formalism of timed automata is a very powerful reference model. Unfortunately it suffers from several undecidability problems, see Alur and Dill (1994), that have been established for some fundamental problems such as checking equality and inclusions of their behaviors (timed languages), checking universality and are not closed under complementation. Hence, there is an increasing interest in studying subclasses of timed automata.

Conceptually much simpler models are weighted automata with weights in semirings like (max,+)-automata. These may be viewed as special timed automata with one clock and only lower bound on clocks. Automata with weights in an interval based semiring then naturally correspond to a more general class of one clock timed automata, where both lower and upper bounds on clocks are allowed. They extend (max,+) automata proposed in Gaubert (1995) as (possibly nondeterministic) weighted automata with weights (multiplicities) in the $(\mathbb{R}\cup\{-\infty\}, \max, +)$ semiring by simply replacing this semiring by a real-interval based semiring, namely, the product of idempotent semirings $\mathbb{R}_{\max}$ and $\mathbb{R}_{\min}$.

Instead of using nondeterminism to code simultaneous execution of events, explicit synchronous composition of deterministic interval weighted automata will be proposed in this paper.

While defining interval weighted automata and their synchronous product, it is important to understand their modeling power and the applied semantics. With this intention, it is pointed out that interval weighted automata are related to Time Petri nets (TPN). TPN are derived from Petri nets by associating holding and/or firing intervals with places and transitions respectively. Because the relation with automata is more explicit, only T-TPN (that is with firing intervals associated with transitions) are considered. These are quite powerful timed models, because their subclass with bounding marking is comparable to timed automata in Cassez and Roux (2008).

Interestingly, the construction proposed in this paper corresponds to a class of timed automata, which is called product interval automata (PIA) in D'Souza and Thiagarajan (2002). They represent a nice trade-off between tractability (fundamental problems like checking timed language inclusion or universality are known to be decidable for this class of timed automata) and modeling power.

It will be shown that the synchronous product of interval weighted automata can be recasted as interval weighted automata over an extended alphabet. The main advantage of our compositional constructions is that the resulting weighted automata models are deterministic, which is not the case for heap automata that capture resource sharing in timed DES using nondeterministic (max,+) automata Gaubert and Mairesse (1999). Nondeterministic weighted automata suffer from many fundamental drawbacks, in particular there is no finite state determinization procedure, no general (state) minimization algorithm is known, and the equality of their behaviors (rational formal power series) is not decidable in general. It is therefore interesting to have deterministic automata representation of timed DES with both synchronization and resource sharing phenomena.

The contribution of this paper is essentially at the modelling level, but it is motivated by control applications. In fact, the behavioral approach (based on formal power series) proposed for the supervisory control of (max,+) automata in Komenda et al. (2009b) can be easily extended to the framework of interval weighted automata. Combined with the synchronous product of interval weighted automata, the decentralized supervisory control of interval

weighted automata is intended to be tackled in future works.

The paper is organized as follows. After preliminaries given in section 2, section 3 is an introduction to interval weighted automata and their algebraic behaviors. In section 4 we discuss closely related discrete event models, namely PIA and T-TPN. In section 5, the definition of the synchronous product of interval weighted automata is proposed with an example. Finally, section 6 proposes a discussion and hints for future investigations.

2. PRELIMINARIES

In this section we recall basic notions and properties from the theory of idempotent semirings (see Baccelli et al. (1992) or Heidergott et al. (2005) for an exhaustive presentation).

2.1 Dioids

Definition 1. An *idempotent semiring* (also called dioid) is a set $\mathcal{D}$ endowed with two inner operations denoted $\oplus$ and $\otimes$. The addition $\oplus$ is commutative, associative, has for unit element ε, *i.e.*, $\varepsilon \oplus a = a$ for each $a \in \mathcal{D}$, and is idempotent: $a \oplus a = a$ for each $a \in \mathcal{D}$. The multiplication $\otimes$ is associative, has for unit element e, and distributes over $\oplus$. Moreover, ε is absorbing for $\otimes$, *i.e.*, $\forall a \in \mathcal{D}: \ a \otimes \varepsilon = \varepsilon \otimes a = \varepsilon$.

Let us recall that any idempotent semigroup (in particular, any dioid) is equipped with a natural order defined by: $a \preceq b \Leftrightarrow a \oplus b = b$.

The simplest examples of dioids are number dioids such as $\mathbb{R}_{\max} = (\mathbb{R} \cup \{-\infty\}, \max, +)$ with idempotent addition $a \oplus b = \max(a, b)$, and conventional addition playing the role of multiplication $a \otimes b$ (or ab when it is unambiguous)$= a + b$.

The direct product of a family of semirings has the structure of a semiring with the operations of addition and multiplication defined componentwise (see for instance (Golan, 1999, chap. 2)). We define below two product dioids resulting from the direct product of $\mathbb{R}_{\max}$ with itself and with $\mathbb{R}_{min}$. The elements of these dioids are pairs which will be used to represent time intervals.

Definition 2. $\mathcal{I}_{max}^{max}$ is the dioid $\langle (R \cup \{-\infty\}) \times (R \cup \{-\infty\}), \oplus, \otimes \rangle$ with:

$$(\underline{p_1}, \overline{p_1}) \oplus (\underline{p_2}, \overline{p_2}) = (\max(\underline{p_1}, \underline{p_2}), \max(\overline{p_1}, \overline{p_2})),$$
$$(\underline{p_1}, \overline{p_1}) \otimes (\underline{p_2}, \overline{p_2}) = (\underline{p_1} + \underline{p_2}, \overline{p_1} + \overline{p_2}),$$

$\varepsilon = (-\infty, -\infty)$ (zero interval) and $e = (0, 0)$ (identity interval).

The dual direct product is defined below.

Definition 3. $\mathcal{I}_{max}^{min}$ is the dioid $\langle (R \cup \{-\infty\}) \times (R \cup \{+\infty\}), \oplus, \otimes \rangle$ with:

$$(\underline{p_1}, \overline{p_1}) \oplus (\underline{p_2}, \overline{p_2}) = (\max(\underline{p_1}, \underline{p_2}), \min(\overline{p_1}, \overline{p_2})),$$
$$(\underline{p_1}, \overline{p_1}) \otimes (\underline{p_2}, \overline{p_2}) = (\underline{p_1} + \underline{p_2}, \overline{p_1} + \overline{p_2}),$$

$\varepsilon = (-\infty, +\infty)$ (zero interval) and $e = (0, 0)$ (identity interval).

In $\mathcal{I}_{max}^{min}$ we have for instance $(2, 4) \oplus (1, 5) = (2, 4)$ and $(2, 3) \oplus (4, 7) = (4, 3)$. Note that dioid $\mathcal{I}_{max}^{min}$ contains imaginary (degenerated) intervals, where lower bound is greater than upper bound.

We propose the following convention for notations. If one of the dioids $\mathcal{I}_{max}^{min}$ or $\mathcal{I}_{max}^{max}$ is fixed, then the dual addition, *i.e.*, addition of the other dioid is denoted by $\oplus'$. This means that in $\mathcal{I}_{max}^{max}$ we have

$$(\underline{p_1}, \overline{p_1}) \oplus' (\underline{p_2}, \overline{p_2}) = (max(\underline{p_1}, \underline{p_2}), min(\overline{p_1}, \overline{p_2}))$$

and *vice versa*.

Matrix dioids are introduced in the same manner as in conventional linear algebra. The identity matrix of $(\mathcal{I}_{max}^{min})^{n \times n}$ (resp. $(\mathcal{I}_{max}^{max})^{n \times n}$) is denoted by E_{max}^{min} (resp. E_{max}^{max}). These identity matrices have zero intervals out of the main diagonal and unity intervals on the main diagonal.

The notation $\mathbb{N}$ is reserved for the set of natural numbers with zero. The star operation can be introduced by the formula

$$a^* = \bigoplus_{n \in \mathbb{N}} a^n,$$

where by convention $a^0 = e$ for any a (for instance, for $A \in (\mathcal{I}_{max}^{min})^{n \times n}$ we have $A^0 = E_{max}^{min}$) and $a^n = a^{n-1} \otimes a$.

Among dioid structures encountered in this paper formal power series and formal languages are very important, because these are behaviors of (interval) weighted automata and corresponding Boolean automata. Formal languages over a finite alphabet A are subsets of the free monoid A^* of all finite sequences of words from A. The zero language is $0 = \{\}$, the unit language is $1 = \{\lambda\}$, where λ is the empty string. In the sequel we will work with the dioid of formal power series in the non commutative variables from A (transition labels) and coefficients from an idempotent semiring $\mathcal{D}$. This semiring will be interval based, but we do not specify it in this section, because firstly the preliminary results below do not depend on particular semiring and secondly, both $\mathcal{I}_{max}^{max}$ and $\mathcal{I}_{max}^{min}$ will be used.

Formal power series with coefficients from a dioid $\mathcal{D}$ and non commuting variables from A form a dioid denoted $\mathcal{D}(A)$, where addition and (Cauchy) multiplication are defined as follows. For two formal power series $s = \oplus_{w \in A^*} s(w)w \in \mathcal{D}(A)$ and $s' \in \mathcal{D}(A)$,

$$s \oplus s' \triangleq \oplus_{w \in A^*} (s(w) \oplus s'(w))w,$$
$$s \otimes s' \triangleq \oplus_{w \in A^*} (\oplus_{uv=w} s(u) \otimes s'(v))w.$$

This dioid is isomorphic to the dioid of generalized dater functions from A^* to $\mathcal{D}$ *via* a natural isomorphism similarly as the dioid $\mathbb{Z}_{\max}(\gamma)$ of formal power series is isomorphic to the dioid of daters from $\mathbb{Z}$ to $\mathbb{Z}_{\max}$, used to study timed event graphs (Baccelli et al., 1992, §5.3). This isomorphism associates to any $y : \ A^* \to \mathcal{D}$ the formal power series $\oplus_{w \in A^*} y(w)w \in \mathcal{D}(A)$.

Finally, we recall basic definitions of tensor linear algebra that will be used in section 5.

The *Kronecker (tensor) product* Graham (1982) $A \otimes^t B$ of matrices $A = (a_{ij}) \in \mathcal{D}^{m \times n}$ and $B \in \mathcal{D}^{p \times q}$ over a dioid, is the $mp \times nq$ block matrix

$$A \otimes^t B = \begin{bmatrix} a_{11} \otimes B & \cdots & a_{1n} \otimes B \\ \vdots & \ddots & \vdots \\ a_{m1} \otimes B & \cdots & a_{mn} \otimes B \end{bmatrix}.$$

3. INTERVAL WEIGHTED AUTOMATA

Interval weighted automata, the basic model of this paper, are introduced as weighted automata with weights in a suitable interval like semiring.

Definition 4. A $\mathcal{D}$-weighted automaton over an alphabet A is a quadruple $G = (Q, \alpha, t, \beta)$, where Q is a finite set of states, $\alpha : Q \to \mathcal{D}$, $t : Q \times A \times Q \to \mathcal{D}$, and $\beta : Q \to \mathcal{D}$, called input, transition, and output delays, respectively.

The (nondeterministic) transition function t associates to a state $q \in Q$, a discrete input $a \in A$ and a new state $q' \in Q$, an output value $t(q, a, q') \in \mathcal{D}$ corresponding to the $a-$transition from q to q' or $t(q, a, q') = \varepsilon$ if there is no transition from q to q' labeled by a.

There are many kinds of weighted automata depending on the multiplicity semiring. For instance, cost or (min,+)-automata with weights in $\mathbb{R}_{min}$ interpreted as costs of transitions, stochastic automata with weights in the probability semirings interpreted as probability of transitions, and (max,+)-automata with weights in $\mathbb{R}_{max}$ interpreted as the exact duration of the transition.

Afterwards, we will consider automata with weights in $\mathcal{I}_{max}^{max}$ which will be called *interval weighted automata*. A value $t(q, a, q') = (\underline{p}, \overline{p})$ of the transition function is then interpreted as follows: $\underline{p}$ (resp. $\overline{p}$) is the minimal (resp. maximal) duration of the a-transition from q to q' (the case where $\underline{p} = -\infty$ and $\overline{p} = -\infty$ corresponds to the case where no a-transition from q to q' exists).

An interval weighted automaton is equivalently defined by a triple (α, μ, β), where $\alpha \in (\mathcal{I}_{\max}^{\max})^{1\times Q}, \beta \in (\mathcal{I}_{\max}^{\max})^{Q\times 1}$ and μ is a morphism defined by:

$$\mu : A \to (\mathcal{I}_{\max}^{\max})^{Q\times Q},\ \mu(a)_{q\,q'} \triangleq t(q, a, q').$$

We will call such a triple a linear representation.

The morphism matrix μ of an interval weighted automaton can be extended from events of A to sequences of A^* using the morphism property

$$\mu(a_1 \dots a_n) = \mu(a_1) \dots \mu(a_n).$$

This means that μ is an element of $\mathcal{I}_{\max}^{\max}(A)^{Q\times Q}$, *i.e.* $\mu = \oplus_{w\in A^*}\mu(w)w$. This way we have a homomorphism from the free monoid of words A^* (with concatenation playing the role of multiplication) to the multiplicative monoid of matrices over the semiring $\mathcal{I}_{\max}^{\max}$.

Let G_1 and G_2 be interval automata defined over local alphabets A_1 and A_2 with $A = A_1 \cup A_2$. The associated natural projections are $P_1 : A^* \to A_1^*$ and $P_2 : A^* \to A_2^*$. We also need the underlying boolean matrices associated to morphism matrices:

$$[B\mu(a)]_{ij} = \begin{cases} e, \text{ if } [\mu(a)]_{ij} \neq \varepsilon, \\ \varepsilon, \quad \text{otherwise.} \end{cases}$$

In order to avoid heavy notation, $B\mu(a)$ is denoted by $B(a)$ in the sequel, hence $B\mu_1(a)$ is denoted by $B_1(a)$. This notation can be extended to words in A^* in an obvious way.

The behavior of an interval automaton $G = (Q, \alpha, t, \beta)$ is given by the formal power series $l(G) : A^* \to \mathcal{I}_{\max}^{\max}$ defined for $w = a_1 \dots a_{n-1} \in A^*$ by

$$l(G)(w) = \max_{q_1,\dots,q_n\in Q} \alpha(q_1) + \sum_{i=1}^{n-1} t(q_i, a_i, q_{i+1}) + \beta(q_n).$$

Thus $l(G)(w)$ is the longest path along the word w starting at an initial state and ending at a final state, which corresponds to the time interval for completion of the sequence of tasks w. This is because $\alpha(q_1) + \sum_{i=1}^{n-1} t(q_{i-1}, a_i, q_i) + \beta(q_n) = \varepsilon$ whenever $\alpha(q_1) = \varepsilon$ (*i.e.*, q_1 is not an initial state) or $\beta(q_n) = \varepsilon$ (*i.e.*, q_n is not a final state). Note that using the linear representation we simply have: $l(G)(w) = \alpha \otimes \mu(w) \otimes \beta$.

4. CONNECTIONS WITH PIA AND T-TPN

In this section two important models studied in the literature will be recalled and compared with interval weighted automata.

4.1 Product Interval Automata

Synchronous composition is a very important concept in the study of large timed automata Alur and Dill (1994). Product interval automata (PIA) are introduced in D'Souza and Thiagarajan (2002) as synchronous products of interval automata (*i.e.* timed automata with a single clock that is always reset after a transition). It is known that PIA are strictly more expressive than interval automata. Consequently, interval automata are not closed under synchronous product.

PIA correspond to an important class of timed automata, where the clocks are read and reset in a particular fashion: there are n clocks (one per component) and during a transition in a PIA only clocks that correspond to the components that are active in a transition are read (*i.e.* compared to constants in guards) and reset. This way the reading and writing (resetting) of clocks is compatible with the distributed event set structure. Thus, the usage of clocks can be completely avoided and PIA can be described by symbolic purely algebraic methods.

When composing interval automata viewed as timed automata, a synchronizing transition is guarded by the intersection of local guards (c.f. definition of the synchronous product of timed automata in Alur and Dill (1994)). In the special case of interval weighted automata this amounts to intersection of corresponding intervals in local automata. This explains why the dual addition of $\mathcal{I}_{\max}^{\max}$ (*i.e.*, addition of $\mathcal{I}_{max}^{min}$) will be needed in our definition of the synchronous product in section 5. The major difference compared to D'Souza and Thiagarajan (2002) is that we are not satisfied with presenting PIA as vectors of interval automata or special class of multi-clock TA described above. Instead, it is shown below how the synchronous product of interval weighted automata can be recasted as interval (weighted) automata over a special extended event set.

4.2 T-time Petri nets

A *Petri net* is a 4-tuple $\mathcal{G} = (\mathcal{P}, \mathcal{T}, \mathcal{F}, M)$, in which $\mathcal{P}$ is a finite set of places, $\mathcal{T}$ is a finite set of transitions,

$\mathcal{F} \subseteq (\mathcal{P} \times \mathcal{T}) \cup (\mathcal{T} \times \mathcal{P})$ is a relation between places and transitions, $M : \mathcal{P} \to \mathbb{N}$ defines the *initial marking* of places [1]. The marking evolves according to the following rules:

(1) Transition t is *enabled* at M if there exists at least one token in each of its input places.
(2) An enabled transition t can fire. The firing of t transforms M into M' by removing one token from each of the input places and adding one token in each of the output places of t.

The set of reachable markings of a Petri net $\mathcal{G}$ will be denoted $\mathcal{R}(\mathcal{G}, M)$ (set of markings reachable from the initial marking M). It can be expressed by its *reachability graph*. Two basic Petri net based models for handling time have been developed: timed Petri nets (see Ramchandani (1974)) and time Petri nets (TPN) (see Berthomieu and Diaz (1991)). With timed Petri nets, a firing finite duration is associated with each transition. With TPN, two values of time $\underline{t}$ and $\overline{t}$ (real or rational numbers) are associated with each transition t: $\underline{t}$ is the minimal time that must elapse, starting from the time at which t is enabled, until this transition can fire; $\overline{t}$ denotes the maximum time during which t can be enabled without being fired. Formally, a TPN is defined as a tuple $(\mathcal{P}, \mathcal{T}, \mathcal{F}, M, T)$ with $T : \mathcal{T} \to \mathbb{R} \times (\mathbb{R} \cup \{+\infty\}), t \mapsto (\underline{t}, \overline{t})$. An important feature to point out in order to complete the semantic adopted for TPN, is how the transition to fire is chosen: afterwards, we consider that decisions on which transitions are to fire is not based on time considerations and that all logically feasible choices can be considered.

In Gaubert and Mairesse (1999), the authors have studied how (max,+) automata are related to timed Petri nets (every *safe* timed Petri net is shown to admit a representation in terms of possibly nondeterministic (max,+) automaton). Considering interval weighted automata in this paper, we naturally study the relation of these models with TPN. More precisely, we explain how to convert a TPN into an interval weighted automaton and discuss the limitations of this approach.

Classically, a Petri net $\mathcal{G}$ can be transformed into an automaton by means of its reachability graph $\mathcal{R}(\mathcal{G}, M)$. Following this approach, the equivalent automaton of a TPN $(\mathcal{P}, \mathcal{T}, \mathcal{F}, M, T)$ is defined an interval weighted automaton (Q, α, t, β) on alphabet A by: $A = \mathcal{T}$, $Q = \mathcal{R}(\mathcal{G}, M)$, $\alpha = M$ and $t(M, t', M') = (\underline{t'}, \overline{t'})$ if t' is enabled by M and its firing transforms M into M'. Note that the obtained automaton is deterministic (since the TPN are assumed to be free-labeled), and that a *safe* TPN has a finite reachability graph (the corresponding automaton is then finite).

Although the above conversion can be done for all TPN, the obtained automaton does not always capture correctly the behavior of the TPN. In fact, it is important to point out that the behavior of the obtained interval weighted automaton, as it is deterministic, is given by:

$$l(G)(w) = \alpha(q_1) + \sum_{i=1}^{n-1} t(q_i, a_i, q_{i+1}) + \beta(q_n).$$

It should be clear that the completion time interval of a sequence of tasks $a_1, \ldots, a_{n-1}$ then corresponds to the sum of completion time intervals of successive tasks. In other words, tasks are considered to be executed sequentially (*i.e.*, tasks cannot be executed simultaneously, even partially). In the proposed model conversion, an event (task) of the automaton corresponds to the firing of a transition. It is then straightforward that the obtained automaton captures correctly the behavior of the TPN only if transitions do not fire simultaneously in the TPN (*i.e.*, no concurrency phemenon occurs).

It could be interesting to identify the largest class of TPN without concurrence, and as by-product, whose behavior could be correctly captured by an interval weighted automaton as defined above. In this paper, we merely consider safe time *state machines* (*i.e.*, Petri nets in which each transition has exactly one input place and one output place) which are intrinsically concurrency free. More precisely, we consider TPN which admit a safe *state-machine covering*, that is, which can be decomposed into *state-machine components* which are synchronized through common transitions. In our approach, each state-machine component is transformed into an interval weighted automaton. To capture the behavior of the whole TPN, interval weighted automata are composed thanks to the synchronous product defined in the following section.

5. SYNCHRONOUS COMPOSITION OF INTERVAL WEIGHTED AUTOMATA

In this section an extension is presented of our synchronous composition of (max,+) automata Komenda et al. (2009a) to more general timed automata (with several clocks).

In particular, the synchronous product of interval weighted automata is still an interval weighted automaton, but over an extended alphabet. Although this alphabet may be infinite, linear representation enables us to compute the behavior on finite sequences. Moreover, if there are no cycles labeled only by private events (i.e. those that are not shared among two or more components), the extended alphabet is finite. This is natural, because a cycle of private events means that the component will never participate in a future synchronization, which is often not acceptable as a behavior of the composed system.

The interval automata with weights in $\mathcal{I}_{max}^{max}$ are considered. They are essentially a sequential model, because the time windows (intervals) in which the events take place follow one after another and simultaneous occurrences of events cannot be modeled in a deterministic manner. This means, that the time window in which an event sequence can take place is the sum of the intervals to which the transition times of the individual events belong, *i.e.* the product of these intervals in $\mathcal{I}_{max}^{max}$. It is our contribution to notice that description of T-time event graphs by systems of linear equations over dioid of real intervals endowed with point wise addition and multiplication; similar to our product semiring $\mathcal{I}_{max}^{max}$) can be extended to more general T-time Petri nets using the synchronous composition.

[1] We restrict our attention to Petri nets which are *ordinary* and *free-labelled* (each transition is labeled by a single event and there are no two transitions with the same label).

The alphabet of the synchronous composition below consists of shared events and of tuples of private local events.

Definition 5. Synchronous Composition of interval weighted automata $G_1 = (Q_1, A_1, \alpha_1, \mu_1, \beta_1)$ and $G_2 = (Q_2, A_2, \alpha_2, \mu_2, \beta_2)$, is the following interval weighted automaton defined over the alphabet $\mathcal{A} = (A_1 \cap A_2) \cup (A \setminus (A_1 \cap A_2))^*$:

$$G_1 \| G_2 = \mathcal{G} = (Q_1 \times Q_2, \mathcal{A}, \alpha, \mu, \beta)$$

with $Q_1 \times Q_2$ the set of states, $\mathcal{A}$ the set of events, $\alpha = \alpha_1 \otimes^t \alpha_2$ the initial delay , $\mu : \mathcal{A}^* \to (\mathcal{I}_{\max}^{\max})^{|Q|\times|Q|}$ the morphism matrix and $\beta = \beta_1 \otimes^t \beta_2$ the final delay. The morphism matrix is defined by :

$$\mu(v) = \begin{cases} \mu_1(v) \otimes^t B_2(v) \oplus' B_1(v) \otimes^t \mu_2(v), \\ \quad \text{if } v = a \in A_1 \cap A_2, \\ \mu_1(P_1(v)) \otimes^t B_2(P_2(v)) \oplus B_1(P_1(v)) \otimes^t \mu_2(P_2(v)), \\ \quad \text{if } v \in (A \setminus (A_1 \cap A_2))^*. \end{cases}$$

For $v \in \mathcal{A}$ such that $v = a \in A_1 \cap A_2$, we have in this case $P_1(v) = P_2(v) = v$ and the expressions for morphism matrix of the composed system are almost the same for both types of events from the extended alphabet. The major difference is that dual addition is used for shared events. This comes from the intuition that synchronization corresponds to intersection of local time constraints (intervals). This way the synchronous composition of interval automata is compatible with the one for more general timed automata, where transitions guards in the synchronous composition are intersections of local transitions guards.

We started with the definition for two local interval automata, where only a (typically finite) part of the events from $\mathcal{A}$ is used. The key point is to find a suitable extended alphabet for any particular composition. It is well known that precise alphabets of local automata play key role in associativity of the synchronous composition.

Let us now explain the intuition behind this seemingly complicated definition. The interval automata G_1 and G_2 are synchronized over the shared events set: $A_1 \cap A_2$, but in between two consecutive synchronizations the automata G_1 and G_2 are free to execute their respective private events belonging to $A \setminus (A_1 \cap A_2)$. The corresponding local strings are $P_1(v)$ and $P_2(v)$. These string are fully independent, hence they are executed simultaneously. Their total duration is then at least the maximum of local lower bounds on duration of component (local) events and at most the maximum of local upper bounds on their duration. Otherwise stated, both upper and lower bounds are given by maximum of the local ones.

On the other hand, for a shared event the situation is different and the dual addition corresponding to the intersection of local intervals appears. This is to ensure compatibility with the synchronous product of timed automata, where time constraints in the global (composed) system are given by intersection of local clock constraints. In our case with one clock components this corresponds to intersection of intervals, which is given by addition of $\mathcal{I}_{max}^{min}$. Note however that if a distributed system is given by a T-time Petri net, then shared transitions have only one time interval attached and it is the same in both components (more generally in all components that share a particular synchronization transition). Otherwise stated, use of $\mathcal{I}_{max}^{max}$ is only sufficient when composing two automata with the same interval duration for shared events (synchronization transitions), which corresponds to the case, where distributed DES is given by a T-TPN, where naturally the transitions shared among two or more components have the same associated interval duration (it is the same transition for all component net that share this transition). In this case our synchronous product is linearly described within $\mathcal{I}_{\max}^{\max}$ and no dual addition is needed.

Still, the dual addition of $\mathcal{I}_{max}^{max}$, *i.e.*, addition of $\mathcal{I}_{max}^{min}$ is useful in modeling of P-time Petri nets. In particular, (interval) P-time event graphs may be modeled by linear recurrent equations in $\mathcal{I}_{max}^{min}$. On the other hand, the semantics of (interval) T-time Petri nets is quite different and T-time event graphs admit linear representation by linear recurrent equations in $\mathcal{I}_{max}^{max}$. This is known for their special class without (even structural) conflicts in transition firings (called T-timed event graphs). The dual addition is then useful even for composing interval automata using the semantics of P-time Petri nets when synchronizing transitions.

5.1 Behavior Induced by Synchronous Composition

In this section the morphism μ of definition 5, extended to $\mu : \mathcal{A}^* \to (\mathcal{I}_{\max}^{\max})^{|Q|\times|Q|}$, will be interpreted in the original alphabet using *induced matrix mapping* ν that is defined on A^* by:

$$\nu(w) = \mu(v_0)\mu(a_1)\mu(v_1)\dots\mu(a_n)\mu(v_n).$$

It is important to remark at this point that the induced matrix mapping $\nu : A^* \to (\mathcal{I}_{\max}^{\max})^{|Q|\times|Q|}$ is not a morphism. It would be a morphism if the synchronous composition could work on the usual global alphabet $A = A_1 \cup A_2$. But this is only possible in the simple case, where $A_1 = A_2 = A$ and clearly $A = \mathcal{A}$.

This concept of induced matrix mapping yields the notion of *induced behavior of the synchronous product*, where behaviors, as formal power series over $\mathcal{A}$, are recasted as formal power series over $A_1 \cup A_2$. It is defined by:

$$l(G_1 \| G_2)(w) = \alpha\nu(w)\beta.$$

Next the induced behavior of the synchronous product of (max,+)-automata G_1 and G_2 is computed.

The formula presented below uses automata representations. Since only deterministic interval automata are considered, there are no problem with canonical (minimal) representations of deterministic formal power series. It turns out that the formula for computing the induced behavior of the synchronous product is quite complicated and complex. In fact, it uses all combinations of morphism matrix and associated logical morphism matrix that are evaluated for projected words (corresponding to sequences of local events). The auxiliary notation below is used.

Let $Z = \{\nu, B\}$ be two elements that can be thought of as Booleans. Thus, we use the complement operation on Z given by $\bar{\nu} = B$ and $\bar{B} = \nu$. The free monoid generated by Z, and denoted as usual by Z^*, is needed. Complement operation can be extended by morphism property to the whole Z^* by defining for $m = m^1 \dots m^k \in Z^*$, $\bar{m} =$

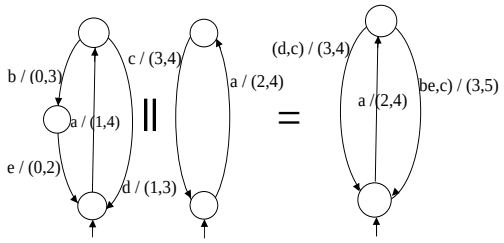

Fig. 1. Automaton representation of G_1, G_2, and $G_1 \| G_2$.

$\bar{m}_1 \dots \bar{m}_k$. La convention est que pour $m \in Z$, m_1 signifie soit ν_1 ou For finite words $m_i = m_i^1 \dots m_i^k \in Z^k, i = 1, 2$ and $v = v_1 \dots v_k \in A^k$ of the same length k we define: $m_i(v) = m_i^1(v_1) \otimes \dots \otimes m_i^k(v_k)$ as a "mixed" product of morphism matrices and their corresponding Boolean matrices. Let us recall at this point that any word $w \in A^*$ can be interpreted as a word of length $2n+1$ over $\mathcal{A}^*$, because w can be decomposed as follows: $w = v_0 a_1 v_1 \dots a_n v_n$, where $a_i \in A_1 \cap A_2$, $i = 1, \dots, n$ are shared events and $v_i \in (A \setminus (A_1 \cap A_2))^*$, $i = 0, \dots, n$ are sequences of private (local) events.

The following results can be shown in the very same way as the formula for induced behavior of the synchronous product of (max,+)-automata from Komenda et al. (2009a).

Proposition 6. The induced behavior of $G_1 \| G_2$ for $w = v_0 a_1 v_1 \dots a_n v_n \in A^*$ can be computed using the formula:

$$l(G_1 \| G_2)(w) = \bigoplus_{m \in Z^{2n+1}} \alpha_1 m_1(P_1(w)) \beta_1 \otimes \alpha_2 \bar{m}_2(P_2(w)) \otimes \beta_2.$$

5.2 Example

We consider interval weighted automata G_1 and G_2 over the alphabets $A_1 = \{a, b, d\}$ and $A_2 = \{a, c\}$, cf. Fig. 1. Their synchronous product, $G_1 \| G_2$ in Fig. 1, is:

$$\mathbf{G_1} \| \mathbf{G_2} = \mathcal{G} = (\mathbf{Q_1} \times \mathbf{Q_2}, \mathcal{A}, \alpha, \mu, \beta),$$

where $Q_1 \times Q_2$ is the set of states,

$$\mathcal{A} = \{a, (be, c), (d, c)\} \subseteq (A_1 \cap A_2) \cup (A \setminus (A_1 \cap A_2))^*,$$

$\alpha = \alpha_1 \otimes^t \alpha_2$, $\beta = \beta_1 \otimes^t \beta_2$, and $\nu(v) =$

$$\begin{cases} \mu_1(a) \otimes^t B_2(a) \oplus B_1(a) \otimes^t \mu_2(a), & \text{if } v = a \in A_1 \cap A_2, \\ \mu_1(be) \otimes^t B_2(c) \oplus B_1(be) \otimes^t \mu_2(c), & \text{if } v = (be, c), \\ \mu_1(d) \otimes^t B_2(c) \oplus B_1(d) \otimes^t \mu_2(c), & \text{if } v = (d, c), \end{cases}$$

Let us note that by definition of morphism we have $\mu_1(be) = \mu_1(b) \otimes \mu_1(e)$ and also $B_1(be) = B_1(b) \otimes B_1(e)$. The extended alphabet is $\mathcal{A} = \{a, (be, c), (d, c)\}$ with $P_1(be, c) = be \in A_1^*$, $P_2(be, c) = c \in A_2^*$, idem for (d, c). The corresponding TPN, drawn in Fig. 2, consists of two composed time state machines.

ACKNOWLEDGEMENTS

This work was supported by the EU.ICT project DISC No. 224498, and by the Academy of Sciences of the Czech Republic Institutional Research Plan No. AV0Z10190503.

6. CONCLUSION

We have shown how techniques based on semirings of interval and tensor linear algebra enable modeling of an important class of timed automata in a linear manner. It is possible to compute behavior of the synchronous product of interval automata based on this linear description.

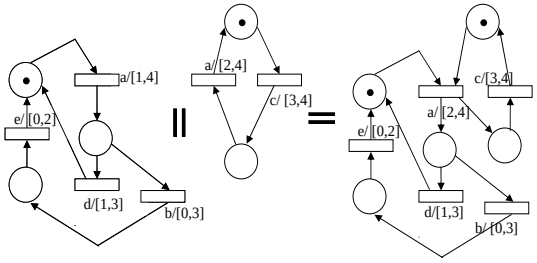

Fig. 2. TPNs corresponding to G_1, G_2, and $G_1 \| G_2$.

The approach presented in this paper together with our earlier results on supervisory control of (max,+) automata paves the way for the decentralized control of (classes of) timed automata like PIA. This is a class of timed automata with all basic problems decidable, but the global control synthesis is of a too high complexity.

REFERENCES

Alur, R. and Dill, D. (1994). The theory of timed automata. *Theoretical computer science*, 126, 183–235.

Baccelli, F., Cohen, G., Olsder, G.J., and Quadrat, J.P. (1992). *Synchronization and Linearity*. J.Wiley & Sons.

Berthomieu, B. and Diaz, M. (1991). Modeling and verification of time dependent systems using time petri nets. *IEEE Transactions on Software Engineering*, 17(3), 259–273.

Cassez, F. and Roux, O.H. (2008). *From Time Petri nets to Timed Automata*. Petri Net: Theory and Application. Edited by Vedran Kordic, I-Tech Publishing, Vienna.

D'Souza, D. and Thiagarajan, P.S. (2002). Product interval automata. *In Sadhana, Academy Proceedings in Engineering Sciences*, 27(2), 181–208.

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

Gaubert, S. and Mairesse, J. (1999). Modeling and analysis of timed petri nets using heaps of pieces. *IEEE Transactions on Automatic Control*, 44(4), 683–698.

Golan, J.S. (1999). *Semirings and their Applications*. Kluwer, Dordrecht.

Graham, A. (1982). *Kronecker Products and Matrix Calculus: With Applications*. J.Wiley and Sons.

Heidergott, B., Olsder, G.J., and Woude, J.V.D. (2005). *Max Plus at Work: Modeling and Analysis of Synchronized Systems*. Princeton Series in Applied Math.

Komenda, J., Lahaye, S., and Boimond, J.L. (2009a). Le produit synchrone des automates (max,+). *Journal Européen des Systèmes Automatisés*, 43, 1033–1047.

Komenda, J., Lahaye, S., and Boimond, J.L. (2009b). Supervisory control of (max,+) automata: a behavioral approach. *Discrete Event Dyn. Systems*, 19, 525–549.

Ramchandani, C. (1974). *Analysis of Asynchronous Concurrent Systems by Timed Petri Nets*. Ph.D. thesis, MIT, Boston.