# Supervisory control of (max,+) automata: extensions towards applications

Sébastien Lahaye, Jan Komenda, Jean-Louis Boimond

# Supervisory control of (max,+) automata: extensions towards applications

*Author version of the article published in*
*International Journal of Control. 2015. Vol. 88 n12 p. 2523-2537*

Sébastien Lahaye, LARIS, Angers, France, sebastien.lahaye@univ-angers.fr
Jan Komenda, Czech Academy of Sciences, Brno, Czech Republic
Jean-Louis Boimond, LARIS, Angers, France

November 2015

**Abstract**

In this paper, supervisory control of (max,+) automata is studied. The synthesis of maximally permissive and just-in-time supervisor, as well as the synthesis of minimally permissive and just-after-time supervisor, are proposed. Results are also specialized to nondecreasing solutions, because only such supervisors can be realized in practice. The inherent issue of rationality raised recently is discussed. An illustration of concepts and results is presented through an example of a flexible manufacturing system.

## 1 Introduction

Discrete event systems (DES) have been extensively studied through several distinct approaches over the past few decades. In particular, a theory initiated by Ramadge and Wonham (Ramadge & Wonham, 1989) lies on the modeling of DES by means of conventional automata; namely, events are seen as letters and DES are seen as finite state machines. The main results concern the logical behavior of DES, and their initial formulations neglected timing aspects. Another approach based on (max,+) algebra considers a more restricted class of DES (Baccelli, Cohen, Olsder, & Quadrat, 1992). In fact, while automata naturally allow the non-determinism inherent to conflicts or choices which often exist in DES (e.g. to capture several possible schedules), (max,+) linear stationary systems are well adapted to DES whose behavior is deterministic (by fixing the schedules in this case). By contrast, timing aspects are natively included and the focus is rather on quantitative measures of the behavior of DES (asymptotic performances, earliest or latest behavior, *etc*).

Stéphane Gaubert first noticed in (Gaubert, 1995) that automata with multiplicities over the (max,+) algebra contribute to fill the gap between these two approaches. Indeed, this last approach is made attractive by the fact that it combines concepts on automata with results on (max, +) algebra, to study at the same time the logical and timing aspects of DES. Recently, (max,+) automata have been applied to performance evaluation (Gaubert, 1995; Gaubert & Mairesse, 1999b), scheduling (Houssin, 2011) and control (Badouel, Bouillard, Darondeau, & Komenda, 2011; Komenda, Lahaye, & Boimond, 2009; Su, van Schuppen, & Rooda, 2012) problems for a large class of timed DES.

This paper deals with control of (max,+) automata and it is then correlated to control approaches for automata and (max,+) linear systems.

On the one hand, *supervisory control* of conventional automata has been proposed as a formal approach to restrict the logical behavior of these systems. It has been extended to timed DES. Let us recall the discrete time approach of (Brandin & Wonham, 1994), where discrete clock is modeled by a special uncontrollable event called tick, and the continuous (dense) time approach based on timed automaton model, where timed automaton is abstracted into logical automaton called region automaton (Wong-Toi & Hoffmann, 1995). It turns out that timed automata form a very general class of timed DES, but many

very basic problems such as checking inclusion of their behaviors (timed languages) is undecidable.

On the other hand, many concepts and results from conventional control theory have been successfully transposed and/or adapted to the (max,+) algebraic setting. Let us mention (without aiming at the exhaustiveness): optimal open-loop control (Cohen, Moller, Quadrat, & Viot, 1989), internal model control (Boimond & Ferrier, 1996), feedback control (Lhommeau, Hardouin, Cottenceau, & Jaulin, 2004), model predictive control (De Schutter & van den Boom, 2001). Nevertheless, these results apply only to the restricted class of (max,+) linear stationary systems, by means of which non-determinism cannot be considered.

The control approach proposed for (max,+) automata thus aims at supplying algebraic results for the control of a wide class of discrete event systems. More precisely, we have developed in (Komenda et al., 2009) a supervisory control framework for (max,+) automata that does not use any abstraction or discretization, and which is based on parallel composition of (max,+) automata with their supervisors. The behavior of this parallel composition is a generalization of Hadamard product that takes into account uncontrollable events (that can neither be prevented from occurring nor delayed).

*Contribution*

In this paper, after recalling our control framework and solutions from (Komenda et al., 2009), we propose new results. First we show how to synthesize a supervisor optimal with respect to a new criterion. Namely, we are interested in minimally permissive and "just-after-time" supervisor in order to guarantee a minimal required behavior and to delay the system as little as possible so that completions of sequences occur later than prescribed dates. We are convinced that this type of control can be useful for applications to transportation networks, manufacturing systems, communication networks, *etc.* For example, in a railway network, one can aim at limiting the number of trains on a path (by increasing dwell times at stations to improve connections) while minimizing the induced delays.

An important aspect towards applicability, namely the rationality of obtained supervisors, had been very partially treated in (Komenda et al., 2009). Based on results from (Badouel et al., 2011), classes of (max,+) automata representing the system and the reference are identified to be sufficient to get rational supervisors.

Another contribution is that results are specialized to the setting of nondecreasing series, which correspond to automata having nonnegative transition values: only such (max,+) automata are of interest when studying DES since transition values correspond to durations in the system.

Each of the concepts and results are illustrated by academic examples through the paper. The final section is devoted to a more substantial example which aims at demonstrating some potential applications to manufacturing systems. A jobshop is considered and the control results are supervisors which restrict the production to a set of possible schedules with deadlines for each of these jobs, and/or dually which impose a minimal production corresponding to mandatory manufacturing orders to fulfill with earliest due dates.

*Scope of applicability*

In the present state of the approach, the system and the control-specification are assumed to be represented by unambiguous (max,+) automata, a sufficient condition to obtain rational supervisors (see discussion in Section 4.5). Recent contributions clarify partially the modeling power of unambiguous (max,+) automata.

Based on results from Gaubert and Mairesse (1999b); Lahaye, Komenda, and Boimond (2014a, 2014b), the focus on unambiguous (max,+) automata leads us to consider timed DES which can be described by safe (or 1-bounded) timed Petri nets[1] for which oriented paths between any two transitions contain at most one "conflict place" (with more than one output transition). In few words, it is then possible to consider timed DES with mono-server resources (i.e. safe) involving synchronization phenomena and several partially asynchronous resource-sharing phenomena. In particular, this framework is general enough to study a wide spectrum of manufacturing systems including safe Flow-shops and Job-shops (as illustrated in Section 5).

A recent study Komenda, Lahaye, and Boimond (2013) has shown that a class of bounded timed

[1] Petri nets are very well known for the modeling of DES (see for example the references (David & Alla, 2010; Murata, 1989) for an introduction).

Petri nets can be represented by deterministic (and thus unambiguous) (max,+) automata. This paves the way for future application of our results to timed DES with multi-server resources (i.e. non-safe).

*Related work*
Besides extensions of supervisory control to timed DES cited above, let us mention the approach in Su et al. (2012) investigating the supervisory control of time-weighted systems by associating to finite-state automata a "mutual exclusion function" as well as a "time-weighted function". This time-weighted function defines for each event a value, interpreted as the duration of the event execution. From the time-weighted system, a heap model (i.e. a particular (max,+) automaton) is built to compute execution times of event sequences in the system. In the corresponding (max,+) automaton, all the transitions involving a given event are assumed have the same weight (one possible duration for an event whatever the current state). Let us point out that, with (max,+) automata, the only way to evaluate execution times of sequences in which an event can occur simultaneously (in an asynchronous manner) with another event is to consider several possible transitions for this event with different durations according to the starting state (see e.g. event $a$ in Fig. 6 whose duration is 2 from state 1 and 0 from state 6 since it is then executed simultaneously with event $c$ leading to this state). In other words, the approach in Su et al. (2012) fits rarely to the class of timed DES (involving parallel activities) considered in this paper.

In Badouel et al. (2011) authors consider the robust control of systems modeled by interval weighted automata. Their control objective is specified by a reference model defining a tolerance on the desired behavior of the plant (instead of a trajectory tracking objective in our case). In addition, their controller differs from our supervisor because it cannot prevent (only delay) the occurrence of events, and no uncontrollable events are considered.

Supervisory control has also been discussed in the framework of time/timed Petri nets. In Buy, Darabi, Lehene, and Venepally (2005) the authors define supervisory controllers for enforcing deadlines on transition firings in time Petri nets. Their goal is to find a supervisor, which can fire a designated transition within a prescribed deadline. In our case, the supervisor can only delay or forbid some events to occur. Furthermore, the events in their setting are associated with firing intervals instead of durations and no uncontrollable transitions are present. In Heidira and Boucheneb (2013) maximally permissive control of time Petri nets is investigated. The time information is then described by intervals instead of durations, and the control problem is about synthesizing a maximally permissive state-based feedback controller (limiting the firing intervals) such that some safety/reachability properties are satisfied. It differs from ours because no time optimality is considered in their paper.

*Organization*
The paper is organized as follows. In the following section we recall important algebraic tools needed in this paper. (Max,+) automata are introduced in section 3, and section 4 is dedicated to their supervisory control. An application to the control of a manufacturing system is presented in section 5. Finally, concluding remarks with hints on future extensions are given in section 6.

# 2 Preliminary definitions and results

The basic algebraic structure used across the paper is that of an idempotent semiring or dioid. In this section, three specific dioids are introduced: (max,+) algebra, the dioid of formal languages and the dioid of formal power series (see the monographs (Baccelli et al., 1992; Heidergott, Olsder, & Woude, 2006) for a more exhaustive presentation). Some results of residuation theory, useful to solve inequalities on such ordered structures, are also mentioned. Finally, basics about projections of languages are reminded.

## 2.1 Dioid algebra

**Definition 1** *A* dioid *is a* semiring *in which the addition $\oplus$ is idempotent. The addition (resp., the multiplication $\otimes$) has a null element $\varepsilon$ (resp., identity element $e$).*

**Example 1** *The set $\mathbb{R} \cup \{-\infty\}$ with the maximum playing the role of addition and conventional addition playing the role of multiplication is a dioid, denoted $\mathbb{R}_{\max}$ and usually called (max,+) algebra, with $e = 0$*

*and* $\varepsilon = -\infty$.
*The set of* $n \times n$ *matrices with coefficients in* $\mathbb{R}_{\max}$*, endowed with the matrix addition and multiplication conventionally defined from* $\oplus$ *and* $\otimes$*, is also a dioid, denoted* $\mathbb{R}_{\max}^{n\times n}$*. The null element for the addition is the matrix denoted* $\varepsilon_n$ *and exclusively composed of* $\varepsilon$ $(= -\infty)$*. We denote* $I_n$ *the identity element of the multiplication, which is the matrix with* $e$ $(= 0)$ *on the diagonal and* $\varepsilon$ $(= -\infty)$ *elsewhere. To be rigorous, note that any* $1 \times n$ *vector should be embedded in this dioid by adding* $n-1$ *lines full of* $\varepsilon$*. To lighten the presentation, this construction is sometimes omitted in the following (without affecting the results), and the coefficients equal to* $\varepsilon$ *in the matrices will be replaced by '·'.*

**Example 2** *If* $A$ *is a finite set (alphabet), the free monoid on* $A$ *is defined as the set* $A^*$ *of finite words with letters in* $A$*. A word* $w \in A^*$ *can be written as a sequence* $w = a_1 a_2 \dots a_p$ *with* $a_1$*,* $a_2$*, …,* $a_p \in A$ *and* $p$ *a natural number.* Formal languages *are subsets of the free monoid* $A^*$*. The set of formal languages, with the union of languages playing the role of addition and concatenation of languages playing the role of multiplication, is a dioid, denoted* $(Pwr(A^*), \cup, .)$*. The zero language is* $\varepsilon = \{\}$*, the unit language is denoted* $e = \{\epsilon\}$ *where* $\epsilon$ *is the empty (zero length) string.*

**Example 3** *A series with coefficients in* $\mathbb{R}_{\max}$ *and indeterminates in free monoid* $A^*$ *is simply a map* $y : A^* \to \mathbb{R}_{\max}$*. The set of these series is denoted* $\mathbb{R}_{\max}[\![A]\!]$*. We will represent a series* $y$ *by a formal sum* $y = \bigoplus_{w\in A^*} y(w)w$*. If* $y(w) = \varepsilon$*, we do not need to write the monomial* $y(w)$ *in the sum. Set* $\mathbb{R}_{\max}[\![A]\!]$ *endowed with point-wise addition and convolution multiplication is a* dioid of formal power series*. Thus, for* $y, y' \in \mathbb{R}_{\max}[\![A]\!]$*, one has:*

$$y \oplus y' \triangleq \oplus_{w\in A^*}(y(w) \oplus y'(w))w \ ,$$

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

*The language* $supp(y) = \{w \in A^* : y(w) \neq \varepsilon(-\infty)\}$ *is called the support of series* $y$*. Another multiplication of formal power series of* $\mathbb{R}_{\max}[\![A]\!]$ *(element-wise or word-by-word), called* Hadamard product*, will be needed and is defined for two series* $y, y' \in \mathbb{R}_{\max}[\![A]\!]$ *by*

$$y \odot y' \triangleq \oplus_{w\in A^*}(y(w) \otimes y'(w))w.$$

*For example, with* $A = \{a, b\}$*,* $y = 1a \oplus 2ab$ *and* $y' = 3ab$*, we have* $supp(y) = \{a, ab\}$*,* $y \oplus y' = 1a \oplus 3ab$*,* $y \otimes y' = 4aab \oplus 5abab$ *and* $y \odot y' = 5ab$*.*

In any dioid, a natural order $\preceq$ is defined by: $a \preceq b \Leftrightarrow a \oplus b = b$. A dioid $\mathcal{D}$ is *complete* if each subset $A$ of $\mathcal{D}$ admits a least upper bound denoted $\bigoplus_{x\in A} x$, and if $\otimes$ distributes with respect to infinite sums. In particular, $\top = \bigoplus_{x\in\mathcal{D}} x$ is the greatest element of $\mathcal{D}$, with the convention $\top \otimes \varepsilon = \varepsilon \otimes \top = \varepsilon$. In a complete dioid, the greatest lower bound $\wedge$ always exists: $a \wedge b = \bigoplus_{x\preceq a, x\preceq b} x$.

Note that natural order of $\mathbb{R}_{\max}$ coincides with usual order and for $y, y' \in \mathbb{R}_{\max}[\![A]\!]$, $y \preceq y'$ (natural order on $\mathbb{R}_{\max}[\![A]\!]$) amounts to $y(w) \leq y'(w)$ for all $w \in A^*$. For example, with $y = 1a \oplus 2ab$, $y' = 3ab$ and $y'' = 1a \oplus 3ab \oplus 2abb$, we have $y \preceq y''$ but $y$ is not comparable to $y'$, $y \wedge y' = 2ab$ and $y \wedge y'' = y$.

Dioid $\mathbb{R}_{\max}$ completed with $\top = +\infty$ is denoted $\overline{\mathbb{R}}_{\max}$, and the dioid of formal power series is complete if coefficients are assumed to belong to $\overline{\mathbb{R}}_{\max}$.

A string $u = u_1 \dots u_k \in A^*$ is called a *subword* of $v \in A^*$ if there exists a factorization $v = v_1 u_1 v_2 \dots v_k u_k v_{k+1}$ with $v_i \in A^*, i = 1, \dots k+1$. The induced subword order on $A^*$ is $u \preceq v$ *iff* $u$ is a subword of $v \in A^*$. For example, *acb* is a subword of *babccbc*. A string $u \in A^*$ is called a *prefix* of $v \in A^*$ if there exists $w \in A^*$ such that $v = uw$. The induced prefix order will also be used: $u \preceq_p v$ *iff* $u$ is a prefix of $v$.

Let $\mathbb{N}$ denote the set of natural numbers with zero. In a complete dioid the star operation (sometimes referred to as *Kleene star*) can be introduced by the formula

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

where by convention $a^0 = e$ and $a^n = a^{n-1} \otimes a$ for any $a$ and $n \geq 1$.

## 2.2 Residuation theory

*Residuation theory* makes it possible to define 'pseudo-inverses' for isotone maps ($f$ is *isotone* if $a \preceq b \Rightarrow f(a) \preceq f(b)$) defined on ordered sets (see (Blyth & Janowitz, 1972), and (Baccelli et al., 1992, §4.4) for the specialization to dioids). Let $\mathcal{B}$, $\mathcal{C}$ and $\mathcal{D}$ be ordered sets. An isotone map $f : \mathcal{D} \to \mathcal{C}$ is said to be *residuated* (resp., *dually residuated*) if $\forall y \in \mathcal{C}$, the least upper bound (resp., greatest lower bound) of set $\{x \in \mathcal{D} \ : \ f(x) \preceq y\}$ (resp., $\{x \in \mathcal{D} \ : \ f(x) \succeq y\}$) exists and belongs to this subset.

**Theorem 1** *Let us denote $Id_{\mathcal{C}}$ and $Id_{\mathcal{D}}$ the identity maps of $\mathcal{C}$ and $\mathcal{D}$. An isotone map $f$ is*

- *residuated iff there exists an isotone mapping $h : \mathcal{C} \to \mathcal{D}$ such that*

$$f \circ h \preceq Id_{\mathcal{C}} \text{ and } h \circ f \succeq Id_{\mathcal{D}}. \tag{1}$$

  *The map $h$ is unique, it is denoted $f^\sharp$ and is called residual of $f$.*

- *dually residuated iff there exists an isotone mapping $h' : \mathcal{C} \to \mathcal{D}$ such that*

$$f \circ h' \succeq Id_{\mathcal{C}} \text{ and } h' \circ f \preceq Id_{\mathcal{D}}. \tag{2}$$

  *The map $h'$ is unique, it is denoted $f^\flat$ and is called dual residual of $f$.*

**Theorem 2** *(Baccelli et al., 1992, §4.4.4) If $f : \mathcal{D} \to \mathcal{C}$ and $g : \mathcal{C} \to \mathcal{B}$ are residuated (resp., dually residuated) mappings, then $g \circ f : \mathcal{D} \to \mathcal{B}$ is also residuated (resp., dually residuated) and $(g \circ f)^\sharp = f^\sharp \circ g^\sharp$ (resp., $(g \circ f)^\flat = f^\flat \circ g^\flat$).*

**Example 4** *The isotone map $R_a : x \mapsto x \otimes a$ in a complete dioid $\mathcal{D}$ is residuated (Baccelli et al., 1992, §4.4). The greatest solution of $x \otimes a \preceq b$ exists and is equal to $R_a{}^\sharp(b)$, also denoted $b \oslash a$.*

**Example 5** *The isotone map $H_y$: $\overline{\mathbb{R}}_{\max}[\![A]\!] \to \overline{\mathbb{R}}_{\max}[\![A]\!]$, $y' \mapsto y' \odot y$ is residuated (Komenda et al., 2009) and its residual is given by*

$$H_y^\sharp(y')(w) = y'(w) \oslash y(w),$$

*i.e., $H_y^\sharp(y') = \bigoplus_{w \in A^*} (y'(w) \oslash y(w))w$. Even more, the Hadamard product admits an inverse, which is known as the Hadamard quotient in the theory of formal power series over rings. However, a generalized version of the Hadamard product, defined and used further in this paper, is only residuated. Hence, the notation from residuation theory is kept also for $H_y$.*

## 2.3 Projection of languages

The *natural projection* from $A$ to $A_c$, where $A_c \subseteq A$ is denoted by $P_c$ and is defined by

$$P_c(a) = \begin{cases} a & \text{if } a \in A_c, \\ \epsilon & \text{if } a \in A \setminus A_c, \end{cases}$$

in which $\epsilon$ denotes the empty string. Projection $P_c$ can be extended to words such that $P_c(a_1 \dots a_n) = P_c(a_1) \dots P_c(a_n)$. It projects away from any word $w \in A^*$ the letters from $A_u = A \setminus A_c$ (natural projection is usually used to remove unobservable events from sequences (Lin & Wonham, 1988)). For a language $L$ (set of words), we have $P_c(L) = \{P_c(w) \ : \ w \in L\}$. The *inverse projection* $P_c^{-1} : \mathrm{Pwr}(A_c^*) \to \mathrm{Pwr}(A^*)$, defined on languages, is such that for all $M \subseteq A_c^*$, $P_c^{-1}(M) = \{w \in A^* \ : \ P_c(w) \in M\}$.

# 3 (max,+) automata

Weighted automata (also known as automata with multiplicities) in the $\mathbb{R}_{\max}$ semiring are usually called *(max,+) automata*. Stéphane Gaubert showed first their advantage to study an important class of timed DES in (Gaubert, 1995). In fact, both synchronization between events and resource sharing (choice) phenomena can be captured by means of (max,+) automata models. In that sense, (max,+) automata

generalize both (max,+)-linear systems (by enabling choice) and logical automata (by introducing duration of events). Moreover, this approach benefits of results both from (max,+)-linear system and supervisory control theories. In this section, we only recall definitions and results which are necessary for the control approach proposed in section 4. The seminal articles (Gaubert, 1995; Gaubert & Mairesse, 1999b) should be consulted for a more exhaustive introduction together with applications to the modeling and performance evaluation of timed DES.

**Definition 2** *A (max,+) automaton $G$ over an alphabet $A$ is a quintuple $G = (Q, A, Q_0, t, Q_m)$, where $Q$ is a finite set of states, $Q_0 \subset Q$ (resp., $Q_m \subset Q$) is the set of initial states (resp., final states) and $t : Q \times A \times Q \to \overline{\mathbb{R}}_{\max}$ is the transition function.*[2]

**Example 6** *Figure 1 is the typical graphical representation which can be associated with every (max,+) automaton:*

- *the nodes correspond to states $q \in Q$;*
- *an arrow exists from state $q$ to state $q'$ if there exists an event $a \in A$ such that $t(q, a, q') \neq \varepsilon$ : it represents the state transition when event $a$ occurs and the value of $t(q, a, q')$ is interpreted as the duration associated to event $a$ (namely, the activation time of event $a$ before it can occur);*
- *an input arrow symbolizes an initial state in $Q_0$;*
- *an output arrow symbolizes a final (or marked) state in $Q_m$.*

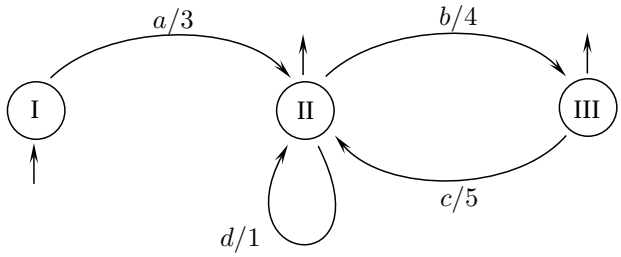


Figure 1: A (max,+) automaton

*For this example, we have $Q = \{I, II, III\}$, $A = \{a, b, c, d\}$, $Q_0 = \{I\}$, $Q_m = \{II, III\}$, $t(I, a, II) = 3$, $t(II, d, II) = 1$, $t(II, b, III) = 4$ and $t(III, c, II) = 5$ (the transition function is equal to $\varepsilon$ for all other triples $(q, a, q')$, $q, q' \in Q$, $a \in A$). This means that the following sequences of events can be generated : a, ab, ad, abc, add, adb, abcb, abcd, addd, addb, adbc, .... More explicitly, the marked language of the underlying logical automaton is $a(d + bc)^*(e + b)$, and its prefix closure gives all the possible sequences.*

In the sequel, the terminology of graph theory is transferred to automata, e.g. to specify path or circuit of an automaton. A path which is both starting with an initial state and ending with a final state is called a *successful path*. The *label* of a transition corresponds to the letter (event) involved in it, and the label of a path is the concatenation of labels of successive transitions. The *weight* of a transition is the value associated by the transition function, and the weight of a path is the $\otimes$ product (i.e. the usual sum) of the weights of the successive transitions.

The *behavior* of (max,+) automaton $G$ is the formal power series $y \in \overline{\mathbb{R}}_{\max}[\![A]\!]$ recognized by this automaton, and is defined by: $\forall w = a_1 \dots a_n \in A^*$,

$$y(w) = \max_{q_0,\dots,q_n \in Q} \left[ \sum_{i=1}^{n} t(q_{i-1}, a_i, q_i) \right], \tag{3}$$

[2]This definition is slightly different from that in (Gaubert, 1995) where initial and final delays are considered. Note that there is no loss of generality here since an automaton with initial and final delays can always be transformed into

where $q_0$ (resp., $q_n$) is an initial state (resp., a final state). Otherwise stated, $y(w)$ is the maximal weight of the successful paths recognizing $w$.

**Remark 1** *The formal power series $y$ is a generalized dater (Gaubert, 1995), where $y(w)$ corresponds to the time instant at which the task sequence $w$ has been completed (by convention $y(w) = -\infty = \varepsilon$ if $w$ does not occur).*

A (max,+) automaton $G$ can also be determined by a triple (known as *linear representation*) $(\alpha, \mu, \beta)$, where

- $\alpha \in \overline{\mathbb{R}}_{\max}^{1\times|Q|}$, $\alpha_q = e$ if $q \in Q_0$ and $\alpha_q = \varepsilon$ otherwise;
- $\mu : A \to \overline{\mathbb{R}}_{\max}^{|Q|\times|Q|}$, $\mu(a)_{q\,q'} \triangleq t(q, a, q')$;
- $\beta \in \overline{\mathbb{R}}_{\max}^{|Q|\times 1}$, $\beta_q = e$ if $q \in Q_m$ and $\beta_q = \varepsilon$ otherwise.

Similarly as timed event graphs can be described by fixed point equations in the dioid of formal power series $\mathbb{Z}_{\max}[\![\gamma]\!]$ (see (Baccelli et al., 1992, §5.3)), any (max,+) automaton can be described by the following equations in $\overline{\mathbb{R}}_{\max}[\![A]\!]$

$$\left\{ \begin{array}{lcl} x & = & x\mu \oplus \alpha, \\ y & = & x\beta, \end{array} \right.$$

in which $\mu = \bigoplus_{a\in A} \mu(a)a \in \overline{\mathbb{R}}_{\max}[\![A]\!]^{|Q|\times|Q|}$ is the morphism matrix. It is known that its least solution, which corresponds to the behavior of the (max,+) automaton, is given by $y = \alpha \otimes \mu^* \otimes \beta$.

**Example 7** *Linear representation of (max,+) automaton $G$ represented in Figure 1 is defined by*

$$\alpha = \begin{pmatrix} e & \cdot & \cdot \end{pmatrix},$$

$$\mu(a) = \begin{pmatrix} \cdot & 3 & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{pmatrix}, \mu(b) = \begin{pmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & 4 \\ \cdot & \cdot & \cdot \end{pmatrix},$$

$$\mu(c) = \begin{pmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & 5 & \cdot \end{pmatrix}, \mu(d) = \begin{pmatrix} \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot \end{pmatrix}, \beta = \begin{pmatrix} \cdot \\ e \\ e \end{pmatrix}.$$

*Let us now compute the behavior of $G$:*

$$\begin{array}{rcl} y & = & \alpha \otimes \mu^* \otimes \beta \\ & = & \alpha \otimes (\mu(a)a \oplus \mu(b)b \oplus \mu(c)c \oplus \mu(d)d)^* \otimes \beta \\ & = & \alpha \otimes \begin{pmatrix} \cdot & 3a & \cdot \\ \cdot & 1d & 4b \\ \cdot & 5c & \cdot \end{pmatrix}^* \otimes \beta. \end{array}$$

*We obtain the following formal power series of $\overline{\mathbb{R}}_{\max}[\![A]\!]$ as the behavior of $G$ :*

$$y = 3a(1d \oplus 9bc)^*(e \oplus 4b).$$

*For instance, $y(ab) = 7$ means that the sequence of events $ab$ is completed at time 7 (it is assumed that the system starts to operate at time 0).*

We end this section with a classification of (max,+) automata (see (Lombardy & Mairesse, 2006)). An automaton is said to be *1-valued* if, for every word $w$, the successful paths labeled by $w$ have the same weight. An automaton is said to be *unambiguous* if, for every word $w$, there is at most one successful path labeled by $w$, in other words, if for every word $w$

1. there exists at most one $i$ such that $(\alpha\mu(w))_i \otimes \beta_i \neq \varepsilon$
2. $\forall a \in A$, $\forall j$, there exists at most one $i$ such that $(\alpha\mu(w))_i \otimes (\mu(a))_{ij} \neq \varepsilon$.

An automaton is said to be *deterministic* if

1. there is a single $i$ such that $\alpha_i \neq \varepsilon$
2. for all $a \in A$, $\forall i$, there exists at most one $j$ such that $\mu(a)_{ij} \neq \varepsilon$.

It should be clear that 'deterministic' implies 'unambiguous' which implies '1-valued'.

# 4 Supervisory control of (max,+) automata

We have recently extended supervisory control paradigm to (max,+) automata in (Komenda et al., 2009). The control problem considered therein is known as maximally permissive (in the logical setting) and optimal with respect to "just-in-time" criterion (in the timed setting). These results are briefly reminded and several contributions are presented in the following subsections. A solution to the dual control problem is derived, namely it is formulated using a minimal required behavior and from the timed viewpoint using "just-after-time" criterion. Both control problems are then specialized to the setting of nondecreasing formal power series, that makes possible the computation of supervisors imposing non negative delays. Finally, rationality of the solutions is discussed.

## 4.1 Principle

The principle of the control proposed for (max,+) automata is close to the one considered for automata in *supervisory control theory* (Ramadge & Wonham, 1989).
The DES is modeled by a (max,+) automaton $G = (Q, A, Q_0, t, Q_m)$ having $y \in \overline{\mathbb{R}}_{\max}[\![A]\!]$ as behavior.
A supervisor interacting with system $G$ is added in order to modify its behavior, that is to restrict the set of recognized sequences and/or to adjust the durations of the sequences which are recognized. For that purpose, the supervisor is able to decide if certain events can occur. More precisely, the partition $A = A_c \cup A_u$ enables us to distinguish

- $A_c$ the set of controllable events: those are events whose validation (and consequently whose occurrence) can be delayed, or even forbidden, by the supervisor;
- $A_u$ the set of uncontrollable events: those are events whose validation can neither be delayed, nor forbidden, by the supervisor.

As represented in Figure 2, the supervisor, denoted $G_s$, interacts as a *feedback* with system $G$. Supervisor $G_s$ is a (max,+) automaton (with $y_s \in \overline{\mathbb{R}}_{\max}[\![A]\!]$ as behavior) defined on the same set of events $A$ which operates in a synchronous manner with system automaton $G$. Each time a state transition occurs in $G$ :

1. supervisor $G_s$ observes event $a \in A$ having occurred in system $G$;
2. its state evolves accordingly;
3. according to the state which has been reached, the supervisor specifies $G_s(a)$ which states the controllable events enabled until the occurrence of a new event in $G$, as well as the validation delay imposed by the supervisor for controllable events[3].

The interaction of supervisor $G_s$ with system $G$ is formalized by means of a *parallel composition* defined in (Komenda et al., 2009) as follows.

**Definition 3** *Let $G_s = (Q_s, A, Q_{s,0}, t_s, Q_{s,m})$ and $G = (Q, A, Q_0, t, Q_m)$ be the (max,+) automata cor-*

an equivalent automaton without such delays by adding new states and by considering these delays as state transitions durations associated to new fictive initial and final events.

[3]Notation $G_s(a)$ is rough because $G_s$ is not specified as a map defined on alphabet $A$.

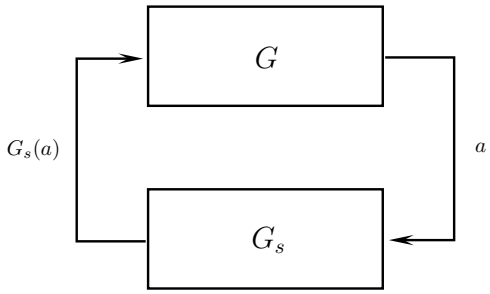


Figure 2: Interaction of supervisor $G_s$ with system $G$ as a feedback.

*responding respectively to the supervisor and the system.* Parallel composition $G_s\|G$ *of* $G$ *and* $G_s$*, which models the system under supervision (also called closed-loop system), is defined by:*

$$G_s\|G = (Q_s \times Q, A, Q_{s,0} \times Q_0), t_{G_s\|G}, Q_{s,m} \times Q_m),$$

*with*

$$t_{G_s\|G}((q_s, q), a, (q'_s, q')) =$$

$$\begin{cases} t_s(q_s, a, q'_s) \otimes t(q, a, q'), & \text{if } a \in A_c, \\ t(q, a, q'), & \text{if } a \in A_u \text{ and } q_s = q'_s, \\ \varepsilon, & \text{if } a \in A_u \text{ and } q_s \neq q'_s. \end{cases} \tag{4}$$

To be convinced that the transition function defined above captures the interaction between the system and the supervisor properly, let us imagine that, at a given time instant, the current state of system $G$ is $q$, the current state of supervisor $G_s$ is $q_s$, and that event $a \in A$ occurs in system $G$:

- For a controllable event $a \in A_c$ and current state $(q_s, q)$ of the closed-loop
  - if there exists a transition associated to this event in $G_s$, (*i.e.* $\exists q'_s$ such that $t_s(q_s, a, q'_s) \neq \varepsilon$), then this event is authorized by the supervisor and its validation in the system is delayed by $t_s(q_s, a, q'_s)$ time units: in the system under supervision, the duration associated with this event is $t_s(q_s, a, q'_s) + t(q, a, q')$;
  - otherwise (*i.e.,* $\nexists q'_s$ such that $t_s(q_s, a, q'_s) \neq \varepsilon$) this event is forbidden by the supervisor: in the system under supervision, the transition function is equal to $t_s(q_s, a, q'_s) \otimes t(q, a, q') = \varepsilon \otimes t(q, a, q') = \varepsilon$, which means that no state transition is possible according to event $a$ (even if such a transition is possible in system $G$).
- For an uncontrollable event $a \in A_u$, the supervisor can not forbid nor delay the occurrence of event $a$. Therefore, the value $t(q, a, q')$ of transition function of the uncontrolled system is kept unchanged in the system under supervision. Moreover, only second component of the state $(q_s, q)$ is updated by firing this uncontrollable event $a \in A_u$, which is described by the last two cases in (4).

## 4.2 Formulations of control problems

In supervisory control approach (see for example (Cassandras & Lafortune, 2006, sec. 3.3)), the specifications, which state the control objectives, are expressed by means of an admissible language. A procedure (an algorithm) derives from this admissible language and from the system language, the supervisor language. An automaton recognizing this language is built to play the role of the supervisor.
Even here, the approach adopted for (max,+) automata is comparable. Indeed, formal power series are considered to express the specified behavior for the system under supervision, and in that sense, they play an equivalent role to the admissible language. These formal power series are jointly used with the ones describing the behavior of the system to compute the behavior (another formal power series) of the supervisor. The supervisor is finally obtained by deriving a (max,+) automaton recognizing this formal

power series as behavior.
It must be clear that the synthesis of the control does not lean on (max,+) automata as objects, but on the formal power series associated with them in order to translate their behavior. Thus let us recall from (Komenda et al., 2009) the behavior resulting from the parallel composition of the supervisor and the system.

**Theorem 3** *The behavior resulting from the parallel composition of $G$ and $G_s$ is given by:*

$$y_{G_s \| G} = y_s \odot_{A_u} y, \tag{5}$$

*in which $\odot_{A_u}$ is called* generalized Hadamard product *and is defined by $\forall w \in A^*$,*

$$(y_s \odot_{A_u} y)(w) \triangleq y_s(P_c(w)) \otimes y(w). \tag{6}$$

The control problem that we consider amounts to solve inequalities in dioid $\overline{\mathbb{R}}_{\max}[\![A]\!]$. More precisely, being given two reference formal power series (bounding the behavior desired for the system under supervision) $y_{ref1}$ and $y_{ref2}$ we want to

$$\begin{array}{c} \text{find the set of series } y_s \text{ such that} \\ y_{ref1} \succeq y_s \odot_{A_u} y \succeq y_{ref2}. \end{array} \tag{7}$$

This problem can be decomposed into two distinct problems, namely:

$$\begin{array}{c} \text{find the greatest series } y_s \text{ such that} \\ y_{ref1} \succeq y_s \odot_{A_u} y, \end{array} \tag{8}$$

and

$$\begin{array}{c} \text{find the smallest series } y_s \text{ such that} \\ y_s \odot_{A_u} y_s \succeq y_{ref2}. \end{array} \tag{9}$$

In fact, the map corresponding to the generalized Hadamard product is isotone, and as we show below solutions, respectively the minimal one for (9) and the maximal one for (8), exist. Then the solution of problem (7) is the interval whose bounds are these extremal solutions. Inequalities (8) and (9) can be interpreted, with the natural order of $\overline{\mathbb{R}}_{\max}[\![A]\!]$ in mind (see example 3), to point out the meaning and the motivations for these control problems:

- Problem (8) consists in finding the greatest formal power series $\overline{y_s}$, that is, the greatest coefficients $\overline{y_s}(w)$ for all words $w$, and thus (since generalized Hadamard product is isotone) the greatest coefficients $(\overline{y_s} \odot_{A_u} y)(w)$, such that $y_{ref1}(w) \succeq (\overline{y_s} \odot_{A_u} y)(w)$. So, the supervisor delays as much as possible the completion of events sequence $w$ within the system under supervision (whose behavior is given by $\overline{y_s} \odot_{A_u} y$). In addition, as $y_{ref1} \succeq y_s \odot_{A_u} y$, the completion date $(\overline{y_s} \odot_{A_u} y)(w)$ in the system under supervision is previous to that specified by $y_{ref1}(w)$, for all $w$. Such a control objective is conform to the so-called *just-in-time* criterion, which has been considered, among others, for the control of timed event graphs (see for example (Baccelli et al., 1992, §5.6), (Lahaye, Boimond, & Ferrier, 2008), (Amari, Demongodin, Loiseau, & Martinez, 2012)). From a logical viewpoint, this supervisor is the most permissive one that restricts the behavior of the system into the maximally allowed one given *via* $y_{ref1}$, where $supp(y_{ref1})$ is known as safety language specification. In the rest of the paper, the solution to this control problem is then referred to as *maximally permissive and just-in-time supervisor*.

- Problem (9) consists in finding the smallest coefficients $\underline{y_s}(w)$ for all $w$, and thus the smallest coefficients $(\underline{y_s} \odot_{A_u} y)(w)$, such that $(\underline{y_s} \odot_{A_u} y)(w) \succeq y_{ref2}(w)$. In other words, the supervisor is then expected to delay as little as possible the system such that the completion date $(\overline{y_s} \odot_{A_u} y)(w)$ in the system under supervision is later than the one specified by $y_{ref2}(w)$, for all $w$. In that sense, it can be compared to the control minimizing delays proposed in (Houssin, Lahaye, & Boimond, 2012) for timed event graphs. Such control problems can also be tackled thanks to model predictive control for max-plus linear systems (De Schutter & van den Boom, 2001). From a logical viewpoint, this supervisor is the least permissive one which guarantees minimal required behavior $y_{ref2}$. Note that similar problems have been considered in supervisory control theory, where minimal required behavior related to the computation of infimal controllable superlanguages has been studied (see for example (Lafortune & Chen, 1990)). From now on, the solution to this control problem is then referred to as *minimally permissive and just-after-time supervisor*.

### 4.3 Maximally permissive and just-in-time supervisor

The notation $H_y^{A_u}$ is used for the operator of generalized Hadamard product, namely

$$H_y^{A_u} : s \mapsto s \odot_{A_u} y. \tag{10}$$

It has been shown in (Komenda et al., 2009) that there is an extension to the result mentioned in example 5 in presence of uncontrollable events ($A_u \neq \emptyset$).

**Proposition 1** *Mapping $H_y^{A_u} : \overline{\mathbb{R}}_{\max}[\![A]\!] \to \overline{\mathbb{R}}_{\max}[\![A]\!]$ is residuated and its residual is defined by:*

$$(H_y^{A_u})^\sharp(y')(w) = \begin{cases} \bigwedge_{u \in P_c^{-1}(w) \cap supp(y)} y'(u) \not/ y(u), & \text{if } w \in A_c^*, \\ T, & \text{if } w \notin A_c^*. \end{cases} \tag{11}$$

**Corollary 1** *The solution to problem (8) is given by*

$$\overline{y_s} = (H_y^{A_u})^\sharp(y_{ref1}).$$

Note that the value $(H_y^{A_u})^\sharp(y_{ref1})(w) = T$ for $w \notin A_c^*$ does not influence the behavior of the system under supervision: in fact, according to formula (6) we only need the values $w \in A_c^*$ of the controller series $(H_y^{A_u})^\sharp(y_{ref1})$ (*i.e.* in the projected words from $A_c^*$).

**Remark 2** *Controllability of the reference series $y_{ref1}$ has been studied in (Komenda et al., 2009). More precisely, we have characterized controllable series as those that can be realized via the control given in Corollary 1 as behaviors of closed-loop systems. In particular we have established a formula for computation of the greatest controllable series that is smaller than the specification series $y_{ref1}$: this series equals $(H_y^{A_u} \circ H_y^{A_u})^\sharp(y_{ref1})$.*

**Example 8** *Let us return to the (max,+) automaton from example 6 and assume that events a, c and d are controllable (can be delayed or even forbidden), while event b is not controllable. Hence, we have the alphabet $A = \{a, b, c, d\}$ with $A_c = \{a, c, d\}$ and $A_u = \{b\}$.*
*We want to impose on the system the following maximal behavior :*

$$y_{ref1} = 7a \oplus 8ad \oplus 9ab \oplus 13abc \oplus 14adb,$$

*which means that sequences a, ad, ab, abc and adb must be achieved at the latest at time instants 7, 8, 9, 13 and 14. Moreover, no other word should be recognized (other possible sequences in the system, such as add or abcd, must be forbidden by the supervisor).*
*By using Corollary1, we obtain $\overline{y_s} = (H_y^{A_u})^\sharp(y_{ref1})$ with :*

$$\overline{y_s}(w) = \top,$$

*for every word $w \notin A_c^*$ (for instance : ab, abc, adb, ...),*

$$\begin{aligned} \overline{y_s}(a) &= y_{ref1}(a) \not/ y(a) \wedge y_{ref1}(ab) \not/ y(ab) \\ &= 7 \not/ 3 \wedge 9 \not/ 7 = 2, \\ \overline{y_s}(ad) &= y_{ref1}(ad) \not/ y(ad) \wedge y_{ref1}(adb) \not/ y(adb) \\ &= 8 \not/ 4 \wedge 14 \not/ 8 = 4, \end{aligned}$$

$$\overline{y_s}(w') = \varepsilon,$$

*for any $w' \in A_c^*$, $w' \notin \{a, ad\}$, in particular*

$$\begin{aligned} \overline{y_s}(ac) &= y_{ref1}(abc) \not/ y(abc) \wedge y_{ref1}(abcb) \not/ y(abcb) \\ &= 13 \not/ 12 \wedge \varepsilon \not/ 16 = \varepsilon. \end{aligned}$$

*The role of the supervisor in this example is to delay the event (a) by two time units, to forbid all occurrences of (c) and to enable the first occurrence of (d) while delaying it by two time units and then to forbid the next occurrences of (d). Note that if the word abc (allowed by the specification $y_{ref1}$) had been enabled, the word abcb could not have been disabled (because b is uncontrollable), which would violate the bound imposed by $y_{ref1}$.*

*The behavior of the controlled system is $\overline{y_s} \odot_{A_u} y = \bigoplus_{w \in A^*} (\overline{y_s}(P_c(w)) \otimes y(w))\, w = 5a \oplus 8ad \oplus 9ab \oplus 12adb$. A (max,+)-automaton having this behavior is represented in Figure 3.*

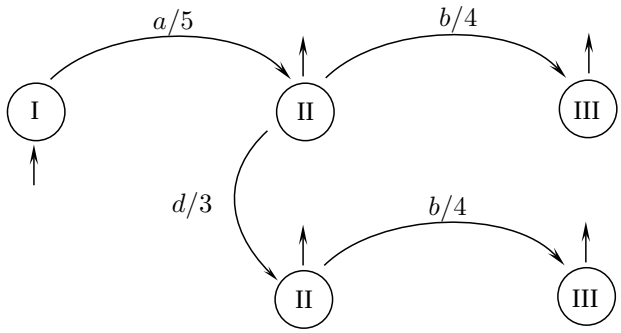


Figure 3: A (max,+)-automaton having $\overline{y_s} \odot_{A_u} y$ as behavior.

### 4.4 Minimally permissive and just-after-time supervisor

In this subsection a solution to the dual control problem is proposed. Formal power series with inverted coefficients defined as follows are used to solve problem (9).

**Definition 4** *For a series $y \in \overline{\mathbb{R}}_{\max}[\![A]\!]$, we denote by $\complement y$ the series of $\overline{\mathbb{R}}_{\max}[\![A]\!]$ defined by $\forall w \in A^*$,*

$$\complement y(w) \triangleq -y(w),$$

*where, by convention, $-\varepsilon = \top$ and $-\top = \varepsilon$. In particular, we have $\complement\complement y = y$.*

It turns out that problem (9) can easily be reformulated as a problem (8) using series with inverted coefficients.

**Lemma 1** *We have*

$$y_s \odot_{A_u} y \succeq y_{ref2} \Longleftrightarrow \complement y_s \odot_{A_u} \complement y \preceq \complement y_{ref2}.$$

**Proof:** We have the following equivalences:

$$\begin{array}{lllll} y_s \odot_{A_u} y & \succeq & y_{ref2} & & \\ \Longleftrightarrow & \forall w, & y_s(P_c(w)) \otimes y(w) & \succeq & y_{ref2}(w) \\ \Longleftrightarrow & \forall w, & y_s(P_c(w)) + y(w) & \geq & y_{ref2}(w) \\ \Longleftrightarrow & \forall w, & -y_s(P_c(w)) - y(w) & \leq & -y_{ref2}(w) \\ \Longleftrightarrow & \forall w, & \complement y_s(P_c(w)) \otimes \complement y(w) & \preceq & \complement y_{ref2}(w) \\ \Longleftrightarrow & & \complement y_s \odot_{A_u} \complement y & \preceq & \complement y_{ref2}. \end{array}$$

□

This implies that the solution computed for (8) can also be used to compute the solution of the dual problem. More precisely, the following corollary holds true.

**Corollary 2** *The solution to problem (9) is given by*

$$\underline{y_s} = \complement\left((H^{A_u}_{\complement y})^\sharp(\complement y_{ref2})\right).$$

*Otherwise stated, $H^{A_u}_y$ is dually residuated and*

$$(H^{A_u}_y)^\flat(y_{ref2}) = \complement\left((H^{A_u}_{\complement y})^\sharp(\complement y_{ref2})\right).$$

**Example 9** *Let us again consider the (max,+) automaton studied in examples 6-8. The control objective of the "dual" control problem is here given by the formal power series $y_{ref2} = 3a \oplus 7ab \oplus 13abc$ that represents the minimal required behavior. Using Corollary 2, we obtain $\underline{y_s} = (H^{A_u}_y)^\flat(y_{ref2}) = \complement\left((H^{A_u}_{\complement y})^\sharp(\complement y_{ref2})\right)$ with :*

- $\underline{y_s}(w) = \varepsilon$, *for any word* $w \notin A_c^*$ *(e.g. ab, abc, ...),*
- $\underline{y_s}(a) = 0$, $\underline{y_s}(ac) = 1$,
- $\underline{y_s}(w') = \varepsilon$, *for any word* $w' \in A_c^*$, $w' \notin \{a, ac\}$.

*The role of the controller in this example is to enable event (a) (without delaying it), to forbid all occurrences of (d) but to authorize the first occurrence of (c) while delaying it by one time unit, and finally to forbid any additional occurrence of (c). The resulting behavior of the controlled system is* $\underline{y_s} = \underline{y_s} \odot_{A_u} y = 3a \oplus 7ab \oplus 13abc \oplus 17abcb$.

**Remark 3** *The above developments could let think that the residuation of Hadamard product can simply be obtained as the Hadamard product with a series with inverted coefficients, that is, series* $H_y^\sharp(y')$ *would be equal to series* $\complement y' \odot y$. *This is, however, not true. For example, if* $y(w) = \varepsilon$ *and* $y'(w) = \varepsilon$ *for a given* $w$, *then we have* $H_y^\sharp(y')(w) = \top$ *whereas* $(\complement y' \odot y)(w) = \varepsilon$.

## 4.5 Rationality of controller series

We have presented formulas to compute the optimal solutions of both control problems (maximally permissive+just-in-time, minimally permissive+just-after-time). However, there is an important issue whether/when the computed series of the controller is rational, because only rational controller series can be realized by a finite (max,+)-automaton. Let us recall that a series is (max,+)-rational (respectively (min,+)-rational) if it can be formed from series with finite support (polynomial series) by using rational operations i.e.: sum $\oplus$ corresponding to max (resp. min), product $\otimes$ and the Kleene star.

The solution $\overline{y_s}$ of control problem (8) given by Corollary 1 is based on residuation of the Hadamard product. It is a well known fact that the Hadamard product is a rational operation recognized by the tensor product of linear representations (see e.g. (Berstel & Reutenauer, 1988)). But the residuation of Hadamard product of rational (max,+) series needs not be (max,+) rational (see e.g. Example 1 in (Badouel et al., 2011)). This is because for a (max,+)-rational series $y$, the series with inverse coefficients $\complement y$ is also (max,+)-rational if and only if $y$ is at the same time (min,+)-rational as shown in (Lombardy & Mairesse, 2006). Nevertheless, Theorem 3.1 in (Badouel et al., 2011) shows that

(a) if $y$ is (min,+) rational and $y'$ is (max,+)-rational, then $(H_y)^\sharp(y')$ is (max,+)-rational

(b) if $y$ is (max,+) rational and $y'$ is (min,+)-rational, then $(H_y)^\sharp(y')$ is (min,+)-rational.

As a corollary, we can identify a sufficient condition for our control problem.

**Corollary 3** *If the behavior* $y \in \overline{\mathbb{R}}_{\max}[\![A]\!]$ *of* $G$ *is (min,+)-rational, then solution* $\overline{y_s} = (H_y^{A_u})^\sharp(y_{ref1})$ *to control problem (8) is (max,+) rational. In addition, if* $y_{ref1} \in \overline{\mathbb{R}}_{\max}[\![A]\!]$ *is (min,+)-rational, then solution* $(H_y^{A_u})^\sharp(y_{ref1})$ *is (max,+) and (min,+)-rational.*

It is known from (Lombardy & Mairesse, 2006) that the class of series that are at the same time (max,+) and (min,+)-rational coincides with the class of unambiguous series (series recognized by an unambiguous automaton). It is also shown that for a (max,+)-rational series $y$, the series with inverse coefficients $\complement y$ is also (max,+)-rational if and only if $y$ is unambiguous. This leads to the following corollary for our second control problem.

**Corollary 4** *If the behavior* $y \in \overline{\mathbb{R}}_{\max}[\![A]\!]$ *of* $G$ *and the reference series* $y_{ref2} \in \overline{\mathbb{R}}_{\max}[\![A]\!]$ *are (min,+)-rational, then solution* $\underline{y_s} = \complement\left((H_{\complement y}^{A_u})^\sharp(\complement y_{ref2})\right)$ *to control problem (9) is (max,+) rational (and unambiguous).*

To conclude, to be able to compute the supervisors for both control problems in an effective way, we will assume that the system-series $y$ and the reference-series are unambiguous.

This assumption requires additional comments since, contrarily to conventional automata, all the (max,+) automata do not admit equivalent deterministic (subsequently unambiguous) models (see (Lombardy & Sakarovitch, 2006)). Let us mention that numerous studies have tackled the determinization problem for automata with multiplicities in a semiring ((Gaubert, 1995; Kirsten, 2008; Klimann,

Lombardy, Mairesse, & Prieur, 2004; Lombardy & Sakarovitch, 2006) is a very partial outline of the literature on this subject). In main lines, semi-algorithms have been proposed, and their termination is guaranteed only for sufficient conditions (to the best of our knowledge, the widest condition is the so-called *clones property* for polynomially ambiguous automata). From the point of view of our control problem for DES, some recent studies have brought important advances. In fact, in the continuity of (Gaubert & Mairesse, 1999b), we have recently contributed to a modeling methodology for DES by means of (max,+) automata (Lahaye et al., 2014a), and identified a significant class which can be represented by deterministic (and thus unambiguous) (max,+) automata (Lahaye et al., 2014b). This class of DES that satisfy the assumptions for our control approach will be introduced in section 5. Moreover, another, language based sufficient condition for determinization of behaviors of timed Petri nets, known as fairness can be imposed by supervisory control as shown in Komenda et al. (2013). This approach can be used for timed Petri nets that do not admit description by deterministic (max,+) automata.

### 4.6 Supervisors with non negative transition weights

It may happen that the solution to problem (8) or (9) is not a non decreasing series. As it will be underlined below, the corresponding supervisor has then negative transition weights, and that constitutes an unrealistic solution. It is then an interesting problem to know whether there exist non decreasing series (that is supervisor with only positive transition weights) solutions to our control problems.

We start with the definition of a non decreasing series from $\overline{\mathbb{R}}_{\max}[\![A]\!]$.

**Definition 5** *A series $s \in \overline{\mathbb{R}}_{\max}[\![A]\!]$ is called* non decreasing on its support *if*

$$\forall v, w \in supp(s), \quad v \preceq_p w \quad \Rightarrow \quad s(v) \preceq s(w), \tag{12}$$

*in which the first inequality employs the prefix order relation on words and the second inequality uses the natural order on $\overline{\mathbb{R}}_{\max}$.*

As an example,

- $y_1 = 1a \oplus 2ab \oplus 1aba \oplus 3abaa$ is not non decreasing on its support because $y_1(ab) \not\preceq y_1(aba)$ ;
- $y_2 = 1a \oplus 2ab \oplus 3abaa$ is non decreasing on its support (although $y(aba) = \varepsilon \prec y(ab) = 2$ because $aba \notin supp(y)$).

**Lemma 2** *The behavior $y$ of an unambiguous (max,+) automaton is a series non decreasing on its support iff all transition-weights are non negative or equal to $\varepsilon$.*

**Proof:** In an unambiguous (max,+) automaton, any $w \in A^*$ is recognized by at most one successful path and operator max is then no more involved in (3). A simple reasoning using mathematical induction can then be used to prove the result.

□

It should be clear that only (max,+)-automata, where all existing transitions have non negative values (or, equivalently, whose formal power series are non decreasing on their supports), can be used as "realistic" supervisors, because the weights of transitions are durations (delays to be imposed on the system). Hence, we are looking for solutions to previously stated control problems in the subset, denoted $\overline{\mathbb{R}}_{\max}[\![A]\!]^{\uparrow}$, of $\overline{\mathbb{R}}_{\max}[\![A]\!]$ that consists of series non decreasing on their support, that is,

$$\begin{array}{c} \text{find } y_s \in \overline{\mathbb{R}}_{\max}[\![A]\!]^{\uparrow} \text{ such that} \\ y_{ref1} \succeq y_s \odot_{A_u} y \succeq y_{ref2}. \end{array} \tag{13}$$

As represented by the commutative diagram of figure 4, we need to investigate if $H_y^{A_u} \circ \mathrm{I}$ is residuated and dually residuated, where I is the canonical injection of $\overline{\mathbb{R}}_{\max}[\![A]\!]^{\uparrow}$ into $\overline{\mathbb{R}}_{\max}[\![A]\!]$. As stated in Theorem 2, this mapping is residuated and dually residuated *iff* $H_y^{A_u}$ and I are. Since by Proposition 1 and Corollary 2 we know that $H_y^{A_u}$ is both residuated and dually residuated, it remains to show that I is residuated

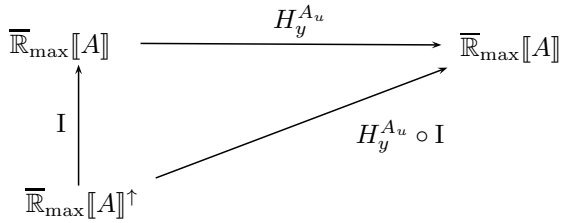


Figure 4: Commutative diagram.

and dually residuated as well.

Notice that we cannot use the term "dioid of series non decreasing on their support" for $\overline{\mathbb{R}}_{\max}[\![A]\!]^{\uparrow}$, because it is not closed under sum of series. For instance, both $s_1 = 1a \oplus 2ab$ and $s_2 = 3a$, are series non decreasing on their support, but their sum $s_1 \oplus s_2 = 3a \oplus 2ab$ is not. Otherwise stated, $\overline{\mathbb{R}}_{\max}[\![A]\!]^{\uparrow}$ is not a sub dioid of $\overline{\mathbb{R}}_{\max}[\![A]\!]$. Still, Propositions stated below show that canonical injection of $\overline{\mathbb{R}}_{\max}[\![A]\!]^{\uparrow}$ into $\overline{\mathbb{R}}_{\max}[\![A]\!]$ is residuated and dually residuated as mapping defined over ordered sets (cf. Theorem 1).

**Proposition 2** *The canonical injection* $\mathrm{I} : \overline{\mathbb{R}}_{\max}[\![A]\!]^{\uparrow} \to \overline{\mathbb{R}}_{\max}[\![A]\!]$ *is residuated and its residual is defined by*

$$\mathrm{I}^{\sharp}(s)(v) = \begin{cases} \bigwedge_{\{w \in supp(s), v \preceq_p w\}} s(w) & \text{if } v \in supp(s), \\ \varepsilon & \text{else.} \end{cases} \tag{14}$$

**Proof:** It is easy to see that $\mathrm{I}^{\sharp}$ defined in (14) is an isotone mapping and also $y_1, y_2 \in \overline{\mathbb{R}}_{\max}[\![A]\!]$, $\quad y_1 \preceq y_2 \quad \Rightarrow \quad supp(y_1) \subseteq supp(y_2)$.
Let us check that I and $\mathrm{I}^{\sharp}$ satisfy the inequalities (1). On one hand we have for $s \in \overline{\mathbb{R}}_{\max}[\![A]\!]$ :

$$(\mathrm{I} \circ \mathrm{I}^{\sharp})(s) = \mathrm{I}^{\sharp}(s) \preceq s.$$

On the other hand, a series non decreasing on its support $s \in \overline{\mathbb{R}}_{\max}[\![A]\!]^{\uparrow}$ satisfies $\forall v, w \in supp(s)$, $v \preceq_p w \quad \Rightarrow \quad s(v) \preceq s(w)$, which implies $s(v) = \bigwedge_{\{w \in supp(s), v \preceq_p w\}} s(w)$. Hence, we obtain

$$(\mathrm{I}^{\sharp} \circ \mathrm{I})(s) = s.$$

□

The series $\mathrm{I}^{\sharp}(s)$ defined by (14) is the greatest series smaller than or equal to $s$ and non decreasing on its support.

**Remark 4** *Proposition 2 constitutes a new formulation of Proposition 5 in (Benveniste, Gaubert, & Jard, 1998) considering prefix order instead of subword order.*

**Example 10** *Let us return to Example 8, with the upper bound specification* $y'_{ref1} = 7a \oplus 8ad \oplus 9ab \oplus 13abc \oplus 14adb \oplus 18abcb \oplus 24abcbc$. *From Corollary 2, we have* $\overline{y'_s} = (H_y^{A_u})^{\sharp}(y'_{ref1})$ *with :*

- $\overline{y'_s}(w) = \top$, *for all word* $w \notin A_c^*$,
- $\overline{y'_s}(a) = 2$, $\overline{y_s}(ac) = 1$, $\overline{y_s}(ad) = 4$,
- $\overline{y'_s}(w') = \top$, *for all word* $w' \in A_c^*$, $w' \notin \{a, ac, ad\}$.

*The series* $\overline{y'_s}$ *is not non decreasing on its support, because* $\overline{y'_s}(a) \not\preceq \overline{y'_s}(ac)$. *Proposition 2 gives us the series non decreasing on its support that is the solution to the maximally permissive and just-in-time control problem. Namely,* $\mathrm{I}^{\sharp}(\overline{y'_s})(a) = 1$, $\mathrm{I}^{\sharp}(\overline{y'_s})(ac) = 1$, $\mathrm{I}^{\sharp}(\overline{y'_s})(ad) = 4$ *and* $\mathrm{I}^{\sharp}(\overline{y'_s})(w') = \top$ *for all* $w' \notin \{a, ac, ad\}$.

We also have the following result.

**Proposition 3** *The canonical injection* $\mathrm{I} : \overline{\mathbb{R}}_{\max}[\![A]\!]^{\uparrow} \to \overline{\mathbb{R}}_{\max}[\![A]\!]$ *is dually residuated and its dual residual defined by*

$$\mathrm{I}^{\flat}(s)(w) = \begin{cases} \bigoplus_{\{v \preceq_p w\}} s(v) & \text{if } w \in supp(s), \\ \varepsilon & \text{else.} \end{cases} \tag{15}$$

**Proof:** The proof follows similar lines as for the residuation (proof of Prop. 2). It is to be noted that

- $(\mathrm{I} \circ \mathrm{I}^{\flat})(s) = \mathrm{I}^{\flat}(s) \succeq s$.
- $s$ non decreasing on its support satisfies $\forall v, w \in supp(s)$, $v \preceq_p w \quad \Rightarrow \quad s(v) \preceq s(w)$, which implies $s(w) = \bigoplus_{\{v \preceq_p w\}} s(v)$, hence $(\mathrm{I}^{\flat} \circ \mathrm{I})(s) = s$.

□

The series $\mathrm{I}^{\flat}(s)$ defined by (15) is the smallest series greater or equal to $s$ and non decreasing on its support.

**Example 11** *Return to Example 9, with specification* $y'_{ref2} = 7a \oplus 9ab \oplus 13abc$. *From Corollary 2, we obtain* $\underline{y'_s} = (H_y^{A_u})^{\flat}(y'_{ref2})$ *with :*

- $\underline{y'_s}(a) = 4$, $\underline{y'_s}(ac) = 1$,
- $\underline{y'_s}(w) = \underline{y_s}(w)$, *for any word* $w \notin \{a, ac\}$.

*The series* $\underline{y'_s}$ *is not non decreasing on its support, because* $\underline{y'_s}(a) \not\preceq \underline{y'_s}(ac)$. *Proposition 3 gives the series non decreasing on its support that is solution to the minimally permissive and just-after-time control problem. More specifically,* $\mathrm{I}^{\flat}(\underline{y'_s})(a) = 4$ *and* $\mathrm{I}^{\flat}(\underline{y'_s})(ac) = 4$.

It follows that for the combined problem the nondecreasing solution is as follows.

**Corollary 5** *The set of solutions to problem (13) is given by the following interval of* $\overline{\mathbb{R}}_{\max}[\![A]\!]^{\uparrow}$

$$\left[\mathrm{I}^{\flat} \circ {H_y^{A_u}}^{\flat}(y_{ref2}), \mathrm{I}^{\sharp} \circ {H_y^{A_u}}^{\sharp}(y_{ref1})\right].$$

**Remark 5** *Let us finally notice that although we do not assume that unambiguous system series* $y$ *and unambiguous specification series* $y_{ref}$ *are non decreasing on their support, it is clear that only such series have practical meaning. Assuming that both series are non decreasing on their support amounts to consider special case of the above proposed solution.*

# 5 Application to the control of a safe Job-shop

Among fields of applications for DES, manufacturing systems are often studied and we focus here on a jobshop system to illustrate our contribution. More precisely, we consider here a jobshop processing two jobs types $\mathcal{J}_1$, $\mathcal{J}_2$.

Job $\mathcal{J}_1$ consists of three elementary tasks $a$, $b$, $c$ which have to be performed according to the sequence $abc$. Processing times of $a$, $b$ and $c$ are all equal to 2.

The production sequence for job $\mathcal{J}_2$ is $def$, with respective durations 1, 1, 3 for elementary tasks $d$, $e$ and $f$.

Two resources $\mathcal{R}_1$ and $\mathcal{R}_2$ can process tasks one by one. Tasks $a$ and $d$ (resp. $c$ and $f$) are processed using resource $\mathcal{R}_1$ (resp. $\mathcal{R}_2$). Both resources $\mathcal{R}_1$ and $\mathcal{R}_2$ are required for tasks $b$ and $e$. We consider all possible sequences with the earliest functioning rule (tasks are completed as soon as possible). We also assume that the system starts operating at date 0.

This jobshop is going to be studied by means of a (max,+) automaton model. Nevertheless, for the sake of clarity, we mention that the system can be modeled by timed Petri net in Figure 5 in which:

- timings are associated with transitions (notation $a/\tau$ means that transition labeled $a$ has $\tau$ for firing time),

- a *preselection policy* is used for conflicts in order to be able to code all the possible choices for production sequences (for a place several output transitions, we consider all possible firings with tokens in this place),
- a token from the initial marking is supposed to have arrived in the Petri net at time instant 0.

Let us point out that several references have studied how to transform a timed Petri net into a (max,+) automaton (and vice-versa). The approaches in (Gaubert & Mairesse, 1999b; Lahaye et al., 2014a) make it possible to associate a (max,+) automaton to any safe timed Petri net such that the completion date of a firing sequence in the Petri net is the same as the one of the corresponding state-transitions sequence in the (max,+) automaton. This (max,+) automaton is generally non-deterministic and may have a larger language than the Petri net (i.e., it recognizes sequences which are not possible firing sequences). A procedure based on completion of heap automata in (Gaubert & Mairesse, 1999a) can be used to determinize such an automaton but it fails for our example: in brief, the cause is that the Petri net includes pairs of transitions without any common input place and any common output place (e.g. $a$ and $f$). The recent contribution (Lahaye et al., 2014a) proposes a recursive procedure which builds a deterministic (max,+) automaton equivalent to a safe timed Petri net. The equivalence corresponds to the facts that the automaton and the Petri net have the same language, and that the completion date of a firing sequence in the Petri net is the same as the one of the corresponding state-transitions sequence in the (max,+) automaton. In addition, it is shown that this procedure terminates if the oriented path between any two transitions contains at most one "conflict-place" (with more than one output transition). This condition is satisfied by the Petri net in Figure 5 and this procedure[4] has been used to obtain (max,+) automaton in Figure 6.

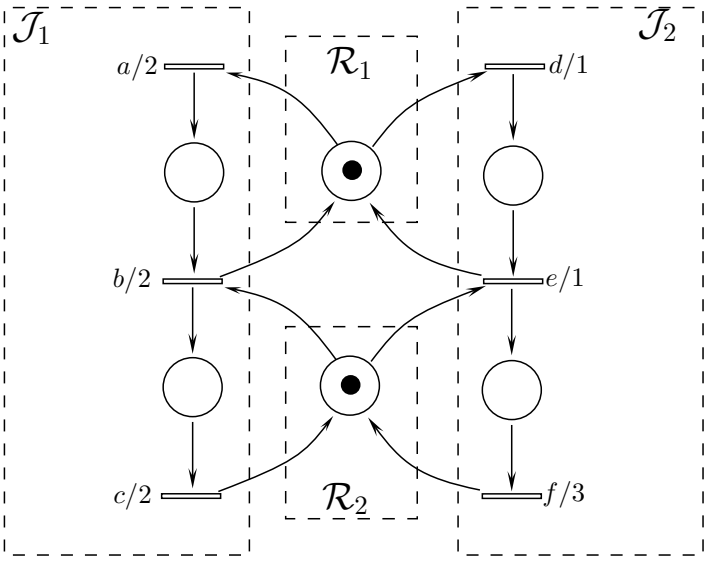


Figure 5: Jobshop represented as a Petri net.

As detailed in section 3 and illustrated in Example 7, it is possible to compute the behavior of the (max,+) automaton in Figure 6 which gives the completion times for all possible production sequences $a_1 a_2 \dots a_n \in \{a, b, c, d, e\}^*$ in the jobshop. For the illustration (and without loss of generality), note that we have chosen only states 6 and 9 as final: these states are reached after the completion of jobs $J_1$ and/or $J_2$ and without other jobs in progress. Among possible scenarios (still without loss of generality), let us also consider production schedules $v_1 v_2 \dots v_m$, in which each production pattern $v_i$, $i = 1, 2, \dots, m$ includes one job $J_1$ and one job $J_2$, that is production schedules satisfying 1/1 ratio for jobs. There are four possible forms for patterns $v_i$: $abcdef$, $abdcef$, $defabc$ and $deafbc$. In fact, task $a$ (resp. $d$) on the one hand, and task $f$ (resp. $c$) on the other hand, are processed in a completely asynchronous manner, which implies all the combinations mentioned in these patterns. Assuming, for example, $A_c = \{a, b, d, e\}$

[4]Please note that the complexity of this procedure remains to be shown.

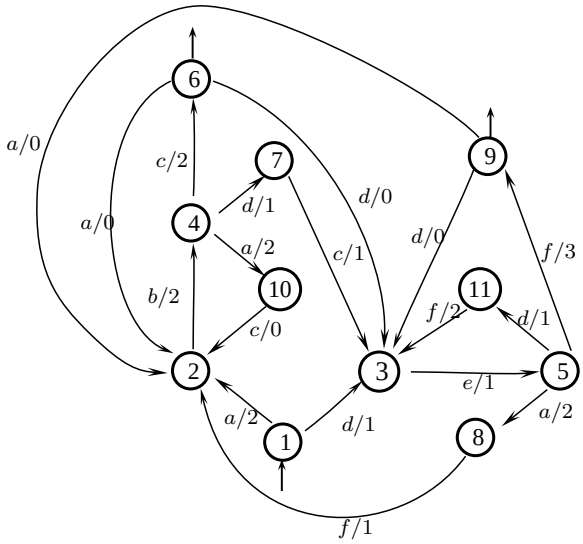


Figure 6: Jobshop represented as a (max,+) automaton.

(i.e. these tasks can be delayed or even disabled) and $A_u = \{b, e\}$, we can use corollary 5 to synthesize supervisors imposing these production schedules with additional logical and timing constraints such as:

(a) A deadline for completing sequences $v_1 v_2 \dots v_i$ equals to $14 \times i$ time units. In order to express this objective, we can use, as maximal required behavior, the following power series in $\overline{\mathbb{R}}_{\max}[\![A]\!]$:

$$y_{ref1} = (14abcdef \oplus 14abdcef \oplus 14defabc \oplus 14deafbc)^*.$$

Note that $y_{ref1}$ is recognized by unambiguous (max,+) automaton depicted in Figure 7.

(b) At the beginning of production, one may be interested in imposing particular setup sequences. For example, let us assume that production has to start with two patterns, each corresponding to sequence $abcdef$ or sequence $defabc$ with 13 units of time as minimal duration. In other words, during at least two cycles, ratio 1/1 must be respected, jobs $J_1$ and $J_2$ must be processed sequentially, and each cycle should be completed in at least 13 time units. In order to translate this objective, we can use as minimal required behavior the following power series in $\overline{\mathbb{R}}_{\max}[\![A]\!]$:

$y_{ref2} = 13abcdef \oplus 13defabc \oplus 26abcdefabcdef \oplus 26abcdefdefabc \oplus 26defabcabcdef \oplus 26defabcdefabc.$

As for $y_{ref1}$, an unambiguous (max,+) automaton recognizing $y_{ref2}$ can easily be found.

**Remark 6** *To the best of our knowledge, there is still no software tool offering the functionalities to perform all the computations to obtain our supervisors. Nevertheless, the Scilab maxplus toolbox*[5]*, which implements various algorithms (sum, product, Kleene star, residuation, etc.) for max-plus matrices, can be used for intermediate steps. There exist several libraries able to manipulate weighted automata (e.g. Vaucanson*[6]*, openFST*[7]*,..), but currently none can be used to implement directly the computations with formal power series to get (max,+) automata supervisors.*

**Remark 7** *Along with supervisory control approaches, our control for (max,+) automata will undoubtedly face a state explosion problem, that is the number of states will tend to grow exponentially in the number of resources and variables in the system. Solutions proposed in logical automata are based on modular, decentralized, hierarchical (abstraction based) control architectures and their combinations, which are*

[5] `http://www.cmap.polytechnique.fr/~gaubert/MaxplusToolbox.html`
[6] `https://www.lrde.epita.fr/wiki/Vcsn`
[7] `http://www.openfst.org`

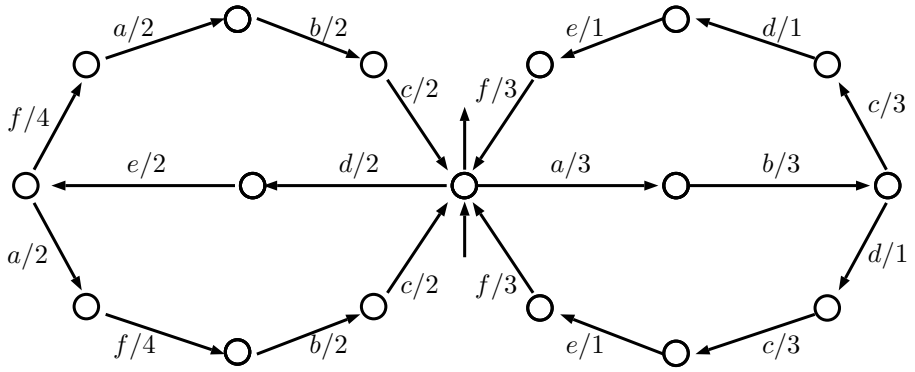


Figure 7: An unambiguous (max,+) automaton recognizing $y_{ref1}$.

*often referred to as heterarchical architectures, which uses both horizontal (decentralized) and (vertical) hierarchical modularity. To be able to deal with large (realistic) systems, these techniques will have to be adapted to (max,+) automata in the future.*

# 6 Conclusion

We have extended the control framework for (max,+) automata so that both maximally permissive and just-in-time as well as minimally permissive and just-after-time supervisors can be synthesized. In order to improve the applicability of results, we have specified classes for which rationality is fulfilled and specialized the results to automata with non-negative transition weights (only these solutions have a realistic meaning). A possible application to flexible manufacturing system has finally been presented as an illustration. Based on composition results from Lahaye et al. (2014a), we plan to extend our results to large systems formed as synchronous products of (max,+) automata representing their subsystems. To be able to deal with state explosion problem, it would also be nice to fit our results into a heterarchical approach. Finally, another possible future investigation is to extend the approach to systems with nondeterministic timing of transitions such as interval weighted automata and their products.

# References

Amari, S., Demongodin, I., Loiseau, J. J., & Martinez, C. (2012). Max-plus control design for temporal constraints meeting in timed event graphs. *IEEE TAC*, *57*(8).

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

Badouel, E., Bouillard, A., Darondeau, P., & Komenda, J. (2011). *Residuation of tropical series: rationality issues* (Rapport de recherche No. 7547). INRIA.

Benveniste, A., Gaubert, S., & Jard, C. (1998). Monotone rational series and max-plus algebraic models of real-time systems. In *Wodes.* Cagliari, Italy.

Berstel, J., & Reutenauer, C. (1988). *Rational series and their languages.* Berlin, Springer Verlag.

Blyth, T. S., & Janowitz, M. F. (1972). *Residuation theory.* Pergamon press.

Boimond, J.-L., & Ferrier, J.-L. (1996, March). Internal Model Control and Max-Algebra: Controller Design. *IEEE Trans. on Automatic Control*, *41*(3), 457-461.

Brandin, B. A., & Wonham, W. M. (1994). Supervisory control of timed discrete-event systems. *IEEE Transactions on Automatic Control*, *39*, 329–342.

Buy, U., Darabi, H., Lehene, M., & Venepally, V. (2005). Supervisory control of time petri nets using net unfolding. In *Proceedings of the 29th annual international conference on computer software and applications conference* (pp. 97–100).

Cassandras, C. G., & Lafortune, S. (2006). *Introduction to des.* Springer-Verlag New York, Inc.

Cohen, G., Moller, P., Quadrat, J. P., & Viot, M. (1989, Jan.). Algebraic tools for the performance evaluation of discrete event systems. *IEEE Proceedings: Special issue on Discrete Event Systems*,

*77*(1).

David, R., & Alla, H. (2010). *Discrete, continuous, and hybrid petri nets (2nd edition)*. Paris: Springer.

De Schutter, B., & van den Boom, T. (2001). Model predictive control for max-plus-linear discrete event systems. *Automatica*, *37*(7), 1049 - 1056.

Gaubert, S. (1995). Performance Evaluation of (max,+) Automata. *IEEE TAC*, *40*(12), 2014-2025.

Gaubert, S., & Mairesse, J. (1999a). Asymptotic analysis of heaps of pieces and application to timed petri nets. In *Proceedings of the 8th international workshop on Petri Nets and performance models* (pp. 158–169). Zaragoza, Spain.

Gaubert, S., & Mairesse, J. (1999b). Modeling and Analysis of Timed Petri Nets using Heaps of Pieces. *IEEE TAC*, *44*(4), 683-698.

Heidergott, B., Olsder, G. J., & Woude, J. V. D. (2006). *Max Plus at work*. Princeton Press.

Heidira, P., & Boucheneb, H. (2013). Maximally permissive controller synthesis for time petri nets. *International Journal of Control*, *86*(3), 1-6.

Houssin, L. (2011). Cyclic jobshop problem and (max,plus) algebra. In *18th ifac wc.* Milan, Italy.

Houssin, L., Lahaye, S., & Boimond, J.-L. (2012). Control Of Constrained (max,+)-Linear Systems Minimizing Delays. *Disc. Event Dyn. Syst.*.

Kirsten, D. (2008). A burnside approach to the termination of Mohri's algorithm for polynomially ambiguous min-plus-automata. *RAIRO - Theoretical Informatics and Applications*, *42*(3), 553-581.

Klimann, I., Lombardy, S., Mairesse, J., & Prieur, C. (2004). Deciding unambiguity and sequentiality from a finitely ambiguous max-plus automaton. *TCS*.

Komenda, J., Lahaye, S., & Boimond, J.-L. (2009). Supervisory Control of (max,+) Automata: A Behavioral Approach. *Disc. Event Dyn. Syst.*, *19*(4), 525-549.

Komenda, J., Lahaye, S., & Boimond, J.-L. (2013). Séquentialisation des réseaux de Petri temporisés. *JESA, special issue including the proceedings of MSR 2013*, *43*(1-3).

Lafortune, S., & Chen, E. (1990). The infimal closed and controllable superlanguage and its applications in supervisory control. *IEEE Transactions on Automatic Control*, *35*, 398–405.

Lahaye, S., Boimond, J.-L., & Ferrier, J.-L. (2008). Just in Time Control of Time-Varying Discrete Event Dynamic Systems in (max,+) Algebra. *International Journal of Production Research (IJPR)*, *46*(19), 5337-5348.

Lahaye, S., Komenda, J., & Boimond, J.-L. (2014a). Compositions of (max,+) automata. *Disc. Event Dyn. Syst.*. (DOI 10.1007/s10626-014-0186-6)

Lahaye, S., Komenda, J., & Boimond, J.-L. (2014b). Modeling of timed petri nets using deterministic (max, +) automata. In *Proceedings of wodes, cachan, france, may 22-24, 2014.*

Lhommeau, M., Hardouin, L., Cottenceau, B., & Jaulin, L. (2004). Interval analysis in diod : Application to robust controller design for Timed Event Graphs. *Automatica*, *40*, 1923-1930.

Lin, F., & Wonham, W. M. (1988). On observability of discrete-event systems. *Information Sciences*, *44*, 173–198.

Lombardy, S., & Mairesse, J. (2006). Series which are both max-plus and min-plus rational are unambiguous. *RAIRO - Theoretical Informatics and Applications*, *40*.

Lombardy, S., & Sakarovitch, J. (2006). Sequential ? *Theoretical Computer Science*, *359*(1-2), 224–244.

Murata, T. (1989, Jan.). Petri Nets : Properties, Analysis and Applications. *IEEE Proceedings: Special issue on Discrete Event Systems*, *77*, 541-580.

Ramadge, P. J. G., & Wonham, W. M. (1989). The control of discrete event systems. *Proceedings of The IEEE*, *77*, 81–98.

Su, R., van Schuppen, J., & Rooda, J. (2012). The synthesis of time optimal supervisors by using heaps-of-pieces. *IEEE TAC*, *57*(1), 105-118.

Wong-Toi, H., & Hoffmann, G. (1995). The control of dense real-time discrete event systems. *Technical Report STA-CS-92-1411, Stanford University*.