Control of \((\max,+)\) Automata: logical and timing aspects

Jan Komenda, Sébastien Lahaye, and Jean-Louis Boimond

Abstract—A new framework for control of \((\max,+)\) automata is introduced. The tensor product of their linear representations and its generalized version used in this paper is an extension of parallel composition from Boolean to \((\max,+)\) automata and can be nicely applied to both logical and timing aspects of supervisory control. Case of uncontrollable events that can neither be disabled nor delayed is studied within a behavioral framework. Optimal (least restrictive) control of \((\max,+)\) automata is studied using residuation theory applied to an extension of Hamadard product of (multivariable) formal power series.

I. INTRODUCTION

\((\max,+)\) automata are weighted automata with weights (multiplicities) in the \((\mathbb{R} \cup \{-\infty\}, \max, +)\) semiring. They have been introduced by S. Gaubert in [10] as a model of Timed Discrete Event (dynamical) Systems (TDES) exhibiting both synchronization of tasks and resource sharing phenomena and moreover can be nondeterministic. \((\max,+)\) automata may also be viewed as a special class of timed automata, a general model for TDES introduced in [1]. Particularly remarkable are the results of [11], where it is shown that \((\max,+)\) automata have a strong expressive power in terms of timed Petri nets: every 1-safe timed Petri net can be represented by special \((\max,+)\) automata, called heap models.

The definition of parallel composition of weighted automata from [2] used in this paper for supervisory control corresponds to the tensor products in terms of linear representation in the \((\mathbb{R} \cup \{-\infty\}, \max, +)\) semiring. The controlled (closed-loop) system is given by the parallel composition of the controller automaton with the plant automaton.

In terms of behaviors, it is Hamadard product of series that is recognized by tensor product of \((\max,+)\) automata (strictly speaking of their linear representations). This means that the behavior (formal power series) of the controlled system is the Hamadard product of behaviors of the controller and the controlled system. Hence, logical and timing aspects of supervisory control can be captured at the same time using our parallel composition. The general case with uncontrollable events that can neither be prevented from happening and can nor be postponed (delayed) is considered in this paper. A generalized version of parallel composition of \((\max,+)\) automata that we propose to take care of uncontrollable events can be expressed using tensor product of the linear representation. Furthermore, in a behavioral (formal power series) framework our parallel composition corresponds to a generalized version (distinguishing uncontrollable events) of Hamadard product. Control with respect to the just in time criterion is then based on the residuation of generalized Hamadard product of formal power series.

The paper has the following structure. In the next section basic algebraic preliminaries needed in the rest of the paper are recalled. In Section III we recall the definition of \((\max,+)\) (weighted) automata and propose a parallel composition of \((\max,+)\) automata, which is applied to control of \((\max,+)\) automata. Section IV is devoted to the main result of the paper: supervisory control of \((\max,+)\) automata with both timing and logical aspects is developed. Conclusion together with hints on future extensions of our approach is given in Section V.

II. ALGEBRAIC PRELIMINARIES

An idempotent semigroup (also called monoid) is a set \(M\) equipped with a commutative, associative inner operation \(\oplus\) that has a unit element \(e\) such that \(e \oplus a = a\) for each \(a \in M\) and satisfies the idempotency condition \(a \oplus a = a\) for each \(a \in M\).

An idempotent semigroup is an idempotent semiring (also called dioid), if it is equipped with another associative inner operation \(\otimes\) that has a unit element \(e\) and that distributes over \(\oplus\), and \(\forall a \in M:\ a \otimes e = e \otimes a = e\). In any dioid, a natural order is defined by: \(a \preceq b \iff a \oplus b = b\). A dioid \(D\) is complete if each subset \(A\) of \(D\) admits a least upper bound denoted \(\bigoplus_{x \in A} x\), and if \(\otimes\) distributes with respect to infinite sums. In particular, \(T = \bigoplus_{x \in T} x\) is the greatest element of \(D\). In a complete dioid, the greatest lower bound, noted \(\wedge\), always exists; \(a \wedge b = \bigoplus_{x \in a, x \leq b} x\). Matrix dioids are introduced in the same manner as in the conventional linear algebra.

The simplest examples of dioids are number dioids such as \(\mathbb{R}_{\text{max}} = (\mathbb{R} \cup \{-\infty\}, \max, +)\) with idempotent addition, denoted by \(\oplus\): \(a \oplus b = \max(a, b)\), and conventional addition playing the role of multiplication, denoted by \(a \otimes b\) (or \(ab\) when unambiguous). If we add \(T = +\infty\) to this set, the resulting dioid is complete and denoted \(\mathbb{R}_{\text{max}}\).

Let us denote by \(\mathbb{N}\) the set of natural numbers with zero. In complete dioids 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 \(a\).

Theorem 2.1 (see [3]): Let \(D\) be a complete dioid, \(x, a, b \in D\). Equation

\[x = x \otimes a \oplus b,\]

admits \(b \otimes a^*\) as least solution.
We recall basic notions and results of residuation theory which allows defining 'pseudo-inverses' of some isotone maps \( f \) is isotone if \( a \leq b \Rightarrow f(a) \leq f(b) \) defined on ordered sets and, in particular, on dioids (see [3], §4.4.4).

**Definition 2.1:** An isotope map \( f : D \to C \), where \( D \) and \( C \) are dioids, is said to be residuated if there exists an isotope map \( h : C \to D \) such that

\[
f \circ h \leq I_D \quad \text{and} \quad h \circ f \geq I_D.
\]

\( I_D \) and \( I_D \) are identity maps of \( D \) and \( C \) respectively. \( h \) is unique, is denoted \( f^\dagger \) and is called residual of \( f \). If \( f \) is residuated then \( \forall y \in C \), the largest upper bound of subset \( \{ x \in D | f(x) \preceq y \} \) exists and belongs to this subset. This greatest subsolution is equal to \( f^\dagger(y) \).

**Theorem 2.2:** In a complete dioid \( D \) the isotope map \( R_a \rightarrow x \cdot x \otimes a \) is residuated. The greatest solution of \( x \otimes a \preceq b \) exists and is equal to \( R_a(b) \), also denoted \( b/a \). This 'quotient' satisfies the following formulæ

\[
(x/a) \otimes a \preceq x, \quad (x \otimes a)/a \preceq x.
\]

Now we recall formal languages, formal power series and their properties. 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 \( 1 = \{e\} \). We say that \( u = u_1 \ldots u_k \in A^* \) is a subword of \( w \in A^\ast \) if there exists a factorization \( w = w_1 u_1 w_2 \ldots u_k w_k w_{k+1} \) with \( u_j \in A^\ast \), \( i = 1, \ldots, k \). The corresponding subword order on \( A^\ast \) is \( u \preceq w \) if \( u \) is a subword of \( w \in A^\ast \).

In the sequel we will work with the dioid of formal power series in the noncommutative variables from \( A \) (transition labels) and coefficients from \( \mathbb{R}_{\max} \) (corresponding to time). Formal power series form a dioid \( \mathbb{R}_{\max}(A) \), where addition and Cauchy multiplication are defined as follows. For two formal power series

\[
s = \oplus_{w \in A^*} s(w)w \in \mathbb{R}_{\max}(A) \quad \text{and} \quad s' = \oplus_{w \in A^*} s'(w)w.
\]

This dioid is isomorphic to the dioid of generalized dater functions from \( A^\ast \) to \( \mathbb{R}_{\max} \) via a natural isomorphism similarly as the dioid \( Z_{\max}(\gamma) \) of formal power series is isomorphic to the dioid of daters from \( Z \) to \( Z_{\max} \) used to study Timed Event Graphs (TEG). This isomorphism associates to any \( y : A^\ast \to \mathbb{R}_{\max} \) the formal power series \( \oplus_{w \in A^*} y(w)w \in \mathbb{R}_{\max}(A) \). This dioid is complete if we work with series that admit coefficients in the completion of \( \mathbb{R}_{\max} \), that is \( \mathbb{R}_{\max} \). We point out that for \( s, s' \in \mathbb{R}_{\max}(A) \), \( s \preceq s' \) with respect to the natural order on \( \mathbb{R}_{\max}(A) \) means that \( \forall w \in A^* : s(w) \preceq s'(w) \) in the sense of natural order on \( \mathbb{R}_{\max} \), i.e. \( s(w) \preceq s'(w) \) for all \( w \in A^\ast \). For instance, with \( A = \{a,b\}, s = 1a \oplus 2ab \) and \( s' = 3ab \), we have \( s \preceq s' = 1a \oplus 3ab \) and \( s \wedge s' = 2ab \). The language \( \text{supp}(s) = \{ w \in A^*: s(w) \neq -\infty \} \) is called the support of the series \( s \). It is known that a formal power series is recognizable by a finite weighted automaton iff it is rational, i.e. it can be formed by rational operations from polynomial series (those with finite support). A formal power series \( s \in \mathbb{R}_{\max}(A) \) is said to be nondecreasing on its support if \( \forall a, w \in \text{supp}(y) : w \preceq u \) implies \( y(w) \preceq y(u) \), where the latter order is the natural order of \( \mathbb{R}_{\max} \) and the former is the subword order on \( A^\ast \) defined above.

Besides Cauchy multiplication of series another multiplication (elementwise or word by word), called Hamamard product, will be needed and is defined by:

\[
s \odot s' = \oplus_{w \in A^*} (s(w) \odot s'(w))w.
\]

The following proposition states that \( H_y : \mathbb{R}_{\max}(A) \to \mathbb{R}_{\max}(A), s \mapsto s \odot y \) is residuated.

**Proposition 2.3:** The isomoting mapping \( H_y : \mathbb{R}_{\max}(A) \to \mathbb{R}_{\max}(A), s \mapsto s \odot y \) is residuated and its residual is given by

\[
H_y^\dagger(s)(w) = (s(w) \odot y(w))w.
\]

\( i.e. H_y^\dagger(s) = \oplus_{w \in A^*} (s(w) \odot y(w))w \).

**Proof:** From definition 2.1, we need to show that \( H_y^\dagger \) is given in (3) is isotone and is such that inequalities (2) are satisfied. Since mapping \( \oplus_x x \odot y \) on \( \mathbb{R}_{\max} \) is isotone, \( H_y^\dagger \) is also isotone. Using successively (f.1) and (f.2) in dioid \( \mathbb{R}_{\max} \), we show the required inequalities:

\[
(H_y \circ H_y^\dagger)(s)(w) = \bigoplus_{w \in A^*} [(s(w) \odot y(w)) \odot y(w)]w = \bigoplus_{w \in A^*} (s(w)w) = s
\]

\[
(H_y^\dagger \circ H_y)(s)(w) = \bigoplus_{w \in A^*} [(s(w) \odot y(w)) \odot y(w)]w = \bigoplus_{w \in A^*} (s(w)w) = s
\]

The notion of left quotient, also known as input derivatives [18], of formal power series is useful. For any series \( s = \oplus_{w \in A^*} s(w)w \in \mathbb{R}_{\max}(A) \) and \( u \in A^\ast \) the series \( u^{-1}s \) is the series with coefficients given by \( a^{-1} = s(w) = s(ww) \) for any word \( w \in A^\ast \). Let us recall from [18] that the derivatives of a formal power series play a role in the construction of a minimal deterministic automaton that recognizes a given formal power series. Since this paper is based on a behavioral approach (i.e. formal power series) these realizations are very important in the supervisor synthesis. On the other hand, from an algebraic point of view there is a characterization of series that are recognizable by (possibly non deterministic) \( \text{max}^+, \text{max}^- \)-automata using this notion of left quotient [16].

Finally, we recall basic definitions of tensor products that will be used in the section III.

If \( A = (a_{ij}) \) is a \( m \times n \) matrix and \( B = (b_{pq}) \) is a \( p \times q \) matrix over a dioid, then their Kronecker (tensor) product \( A \otimes B \) is the \( mp \times nq \) block matrix

\[
A \otimes B = \begin{bmatrix}
a_{11} \otimes B & \cdots & a_{1n} \otimes B \\
\vdots & \ddots & \vdots \\
a_{m1} \otimes B & \cdots & a_{mn} \otimes B
\end{bmatrix}
\]
III. PARALLEL COMPOSITION OF (max,+ ) AUTOMATA

First we recall the definition of (max,+ ) automata, which are automata with multiplicities in the \(\mathbb{R}_{\text{max}}\) semiring. The formal definition of maxplus (weighted) automata is taken from [10].

**Definition 3.1:** A (max,+ ) automaton over an alphabet \(A\) is a quadruple \(G = (Q, \alpha, t, \beta)\), where \(Q\) is a finite set of states, \(\alpha : Q \to \mathbb{R}_{\text{max}}, \ t : Q \times A \times Q \to \mathbb{R}_{\text{max}}, \) and \(\beta : Q \to \mathbb{R}_{\text{max}}\), called input, transition, and output delays, respectively.

The transition function associates to a state \(q \in Q\), a discrete input \(a \in A\) and a new state \(q' \in Q\), an output value \(t(q, a, q') \in R\) corresponding to the \(a\)-transition from \(q\) to \(q'\) or \(t(q, a, q') = \varepsilon\) if there is no transition from \(q\) to \(q'\) labelled by \(a\). The real output value of a transition is interpreted as the minimal duration of the transition.

A (max,+ ) automaton is equivalently defined by a triple \((\alpha, \mu, \beta)\), where \(\alpha \in \mathbb{R}_{\text{max}}^{|Q|}, \beta \in \mathbb{R}_{\text{max}}^{|Q|}\) and \(\mu\) is a morphism defined by:

\[
\mu : A \to \mathbb{R}_{\text{max}}^{Q \times Q}, \quad \mu(a)_{q,q'} \triangleq t(q, a, q') .
\]

We will call such a triple a linear representation.

Note that the morphism matrix \(\mu\) of a (max,+ ) automaton can also be considered as element of \(\mathbb{R}_{\text{max}}(A)^{|Q| \times |Q|}\), i.e. \(\mu = \bigoplus_{a \in A} \mu(a) a\). We associate to states of \(G\) variables (formal power series) \(x_2, x_1, x_3 \in \mathbb{R}_{\text{max}}(A)\) from left to right. We obtain the following equations in dioid \(\mathbb{R}_{\text{max}}(A)\) endowed with pointwise addition and convolution multiplication:

\[
\begin{align*}
x_1 &= x_2 \otimes (2b) \oplus x_3 \otimes (4d) \oplus e \\
x_2 &= x_1 \otimes (1a) \\
x_3 &= x_1 \otimes (3c) \\
y &= x_1 \oplus x_2 \oplus x_3 .
\end{align*}
\]

In terms of matrices we obtain:

\[
(x_1 \ x_2 \ x_3) = (x_1 \ x_2 \ x_3) \begin{pmatrix} \varepsilon & 1a & 3c \\ 2b & \varepsilon & \varepsilon \\ 4d & \varepsilon & \varepsilon \end{pmatrix} \oplus \alpha
\]

\[
y = x \beta ,
\]

where \(x = (x_1 \ x_2 \ x_3)\), \(\alpha = (e \ e \ v)\), and \(\beta = (e \ e \ e)^T\).

In general we have the following linear description of (max,+ ) automata in the dioid \(\mathbb{R}_{\text{max}}(A)\) of formal power series:

\[
\begin{align*}
x &= x \mu \oplus \alpha \\
y &= x \beta,
\end{align*}
\]

where we also call \(\mu = \bigoplus_{a \in A} \mu(a) a \in \mathbb{R}_{\text{max}}(A)\) the morphism matrix.

Recall that according to theorem 2.1 the least solution to this equation is \(y = \alpha \mu^* \beta\).
We have proposed supervisory control framework for (max,+) automata using an operation called supervised composition in [13] and automata, i.e. state based framework. In the meantime we have noticed that a similar operation has been defined by a kind of structural induction for formal power series in [7]. However the control problems with this product could not be solved efficiently and in the full generality unless logical (supervisory control) aspect is separated from the timing aspect by giving an a priori logical structure describing a subset of possible words (schedules). This is because of the definition of supervised product proposed in [13] is more complicated than usual definition of parallel composition of weighted automata (cf. Arnold [2]). Moreover, within the state based framework there are problems with residuability of certain mapping.

The parallel composition below is defined as an extension of parallel composition (synchronous product) from logical problems with residuability of certain mapping.

Their parallel composition is

\[ G_c = (Q_c, q_c, 0, Q'_m, t_c), \quad G = (Q_g, q_g, 0, Q'_g, t_g). \]  

(6)

Their parallel composition is

\[ G_c ||_{A_e} G = (Q_c \times Q_g, q_0, Q_m, t) \]

with \( q_0 = \langle q_c, 0, q_g, 0 \rangle \), \( Q_m = Q'_c \times Q'_g \),

\[ t = \begin{cases} t_c(q_c, a, q'_c) \otimes t_g(q_g, a, q'_g), & \text{if } a \in A_e, \\ t_g(q_g, a, q'_g), & \text{if } a \in A_u \end{cases} \]  

(7)

The interpretation of this definition (that can also be viewed as an extension of prioritized synchronous composition from [12] or [15] from Boolean to the (max,+) case) should be clear.

Controllable transitions (i.e. \( t_g(q_g, a, q'_g), \ a \in A_e \)) in the plant \( G \) can be in the composed system \( G_c ||_{A_e} G \) both disabled (due to \( \varepsilon \) absorptive for multiplication : when the synchronizing transition of the controller is not defined \( t_c(q_c, a, q'_c) = \varepsilon \) and delayed (when \( t_c(q_c, a, q'_c) \geq 0 \)). The delay is added to the duration of the corresponding transition in \( G_c ||_{A_e} G \). On the other hand, uncontrollable transitions (i.e. \( t_g(q_g, a, q'_g), \ a \in A_u \)) in the plant \( G \) can be in the composed system \( G_c ||_{A_e} G \) neither disabled nor delayed.

Remark 3.1: There is the following interpretation of the parallel composition of a system with its controller. The controller is another (max,+) automaton running in parallel (in a standard synchronous manner) with the system’s automaton, that observes the generated events and either generates the same event as the controller, in which case it may delay the execution of the corresponding transition by the number of time units given by the weights of the transition in the controller (in case of a controllable event) or does not generate this event. In the latter case the event that was possible in the uncontrolled system is disabled in the parallel composition (this event should be controllable in accordence with definition). Uncontrollable events can neither be prevented from happening and can nor be delayed, the uncontrollable transition in the parallel composition inherits the duration from the original uncontrolled plant \( G \).

It is easy to see that the following holds.

**Proposition 3.2:** Consider two (max,+) automata and their linear representations:

\[ G_c = (\alpha_c, \mu_c, \beta_c), \quad G = (\alpha_g, \mu_g, \beta_g). \]  

(8)

Their parallel composition in terms of linear representations is

\[ G_c ||_{A_e} G = (\alpha_t, \mu_t, \beta_t) \]

\[ \alpha_t = \alpha_c \otimes^T \alpha_g, \]

\[ \forall a \in A_e : \mu(a) = \mu_c(a) \otimes^T \mu_g(a), \]

\[ \forall a \in A_u : \mu(a) = E \otimes \mu_g(a), \]

\[ \beta_t = \beta_c \otimes^T \beta_g. \]

**Proof:** The proof of this proposition is quite simple and follows from the definition of tensor multiplication and graphical interpretation of morphism matrices. Let us consider first the case \( a \in A_e \). It must be shown that

\[ \mu(a) = \mu_c(a) \otimes^t \mu_g(a), \]

i.e. that

\[ [\mu(a)]_{ik,jl} = [\mu_c(a)]_{ij} \otimes [\mu_g(a)]_{kl}. \]

According to the graphical interpretation of morphism matrix \( [\mu(a)]_{ik,jl} \) is the weight associated to \( a \)-transition from the state labelled by \( ik \) to the state labelled by \( jl \) of \( G_c ||_{A_e} G \).

According to definition 3.2 it should be equal to the product \( \otimes \), i.e. usual sum of the weights of the transition from the state labelled by \( i \) to the state labelled by \( j \) of the controller \( G_c \) and the transition from the state labelled by \( k \) to the state labelled by \( l \) of the plant \( G \). Hence,

\[ [\mu(a)]_{ik,jl} = [\mu_c(a)]_{ij} \otimes [\mu_g(a)]_{kl}. \]

follows from definition 3.2 and from the very definition of tensor product

\[ \mu(a) = \mu_c(a) \otimes^t \mu_g(a). \]

The case \( a \in A_u \) is even simpler.

We will show that \( \mu(a) = E \otimes \mu_g(a) \), which according to the definition of tensor product can be rewritten as

\[ [\mu(a)]_{ik,jl} = E_{ij} \otimes [\mu_g(a)]_{kl}. \]

The (max,+) identity matrix is given by

\[ E_{ij} = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases} \]

The graphical interpretation of \( [\mu(a)]_{ik,jl} \) is the weight associated to \( a \)-transition from the state labelled by \( ik \) to the state labelled by \( jl \) of \( G_c ||_{A_e} G \). According to definition
3.2 it should be equal to the weight of the $a$-transition from the state labelled by $k$ to the state labelled by $l$ of $G$, i.e. to $\{\mu_l(a)\}_{k,l}$. This is exactly expressed by the formula $\mu(a) = E \otimes \mu(a)$. Using once again definition 3.2, this should be equal to the weight of the transition from the state labelled by $k$ to the state labelled by $l$ of $G$. This corresponds exactly to the term $E_{ij} \otimes (\mu_l(a))_{ji}$. Concerning the initial and final delays, it is obvious that $\alpha_{ij} = (\alpha_c)_i \otimes (\alpha_g)_j$, i.e. $\alpha = \alpha_c \otimes T \alpha_g$ expresses exactly the fact the composed state is initial iff its component are initial states of $C$ and $G$ and similarly for $\beta$.

Proposition 3.2 is useful for computing the behavior of the composed system consisting of a controller and a plant. Although we have formulated parallel composition in the state based framework (in order to make a clear connection with the classical supervisory control theory) the last proposition can be viewed as an equivalent definition of parallel composition for $(\max,+)$ automata in terms of their linear representations that admit nonzero initial and final delays from $\mathbb{R}_{\max}$.

IV. APPLICATION TO SUPERVIZY CONTROL

In this section, parallel composition of Definition 3.2 is applied to the supervisory control of $(\max,+)$ automata. The aim is to satisfy at the same time both qualitative (logical) and a quantitative (e.g. timing) specification given by a $(\max,+)$ automaton. The controlled (closed-loop) system is then simply the parallel composition of a supervisor to be found with the plant. The supervisor is itself represented by a $(\max,+)$ automaton.

We recall that the common event alphabets of the system and the controller is $A$. As usual in supervisory control, $A = A_c \cup A_u$ is partitioned into disjoint subsets of controllable events (which can be forbidden and delayed) and uncontrollable events (which can neither be forbidden nor delayed).

A behavioral approach is adopted: we work with the dioid of formal power series $\mathbb{R}_{\max}(A)$ instead of with $(\max,+)$ automata themselves. This means we are interested in synthesizing the formal power series of the controller (denoted by $y_c$) that enforces both logical and timing specification given by a formal power series $y_{ref} \in \mathbb{R}_{\max}(A)$, recognized by a $(\max,+)$ automaton $G_{ref}$. The final step of the supervisor synthesis is then to construct a supervisor with the behavior given $y_C \in \mathbb{R}_{\max}(A)$. This can be done by standard methods known from the literature. In the simplest case the minimal deterministic supervisor $G_c$ that recognizes $y_c$ is constructed (e.g. methods of universal coalgebra [18]). More involved is the construction of a (potentially smaller) nondeterministic supervisor recognizing $y_C$ (cf. [16]).

Let us now detail the considered control problem and its solution. We start with the simpler case $A_c = A$ (no uncontrollable event). Given a specification behavior $y_{ref} \in \mathbb{R}_{\max}(A)$, the considered problem is to find the greatest "controller" series, denoted $y_C \in \mathbb{R}_{\max}(A)$ such that
\[
y_C \circ y \preceq y_{ref}.
\] Solving this equation in dioid $\mathbb{R}_{\max}(A)$ is a residuation problem solved by proposition 2.3: the greatest solution of (9) exists and is given by
\[
y_C^{opt} := H^*_y(y_{ref}).
\]

Having in mind the meaning of order relation in $\mathbb{R}_{\max}(A)$, one can give the following interpretations. Finding the greatest $y_C^{opt}$, that is the greatest coefficients $y_C^{opt}(w)$ for all $w$, and as a by-product the greatest coefficients $y_C^{opt}(w) \otimes y$, means that the controller will delay as much as possible the completion of the sequence of events traduced by $w$ in the supervised system (whose behavior is given by $y_C^{opt} \otimes y$). In addition, since $y_C^{opt} \otimes y \preceq y_{ref}$, the completion date in the supervised system $y_C^{opt}(w) \otimes y$ is earlier than the completion date specified by $y_{ref}(w)$ for all sequence $w$.

In other words, the considered control objective satisfies the just-in-time criterion, notably considered for the control of TEGs (see §5.6 in [3]).

In the general case with $A_c \subset A$, i.e. $A_u \neq \emptyset$, a natural approach is to synthesize the supremal controllable sub-language of the logical specification (language associated with $y_{ref}$) using standard methods of supervisory control (see [19]). Let us denote $y_C^\dagger$, the restriction of the specification to its supremal controllable sub-language. Referring to equation (10), the behavior of the desired controller is given by
\[
y_C^{opt} := H^*_y(y_{ref}).
\]

As mentioned earlier, the final step of the control synthesis consists of finding a $(\max,+)$ automaton realization of the controller formal power series $H^*_y(y_{ref})$ (or $H^*_y(y_C^\dagger)$).

A more challenging approach we investigate is to handle the logical and timing part of the specification at the same time.

The following relationship between tensor product and usual product of matrices, well known as the mixed product property will be useful:

Property 4.1: For matrices $A, B, C, D$ of suitable dimensions we have:
\[(A \otimes C) \otimes (B \otimes D) = (A \otimes B) \otimes (C \otimes D)\]

Our aim now is to find a formula for behavior (i.e. formal power series) of parallel composition of the controller $(\max,+)$ automaton with the plant $(\max,+)$ automaton. Let us denote $P_c$ the natural projection from $A^*$ to $A_c^*$ that is morphism of monoids that from any string $w \in A^*$ projects away the uncontrollable events from $A_u$, cf. [17]. Natural projections have many useful properties, among them we need only

Lemma 4.2: Let $A_c \subseteq A$ with the corresponding natural projection $P_c : A^* \rightarrow A_c^*$ and the inverse projection $P_c^{-1} : \text{Pwr}(A_c^*) \rightarrow \text{Pwr}(A^*)$. Then

(i) $P_c \circ P_c^{-1}$ is identity, i.e. $\forall L \subseteq A^*$ : $P_c(P_c^{-1}(L)) = L$

(ii) $\forall L \subseteq A^*$ : $L \subseteq P_c^{-1}(P_c(L))$

Proof: This fact is very well known and useful in supervisory control (cf. [17]) and follows from abstract algebra, because $P_c$ is a surjective map. The inclusion in
Using Lemma 4.3, we have the following theorem.

**Theorem 4.4:**

\[
\begin{align*}
\{ \mu_a(a) \otimes \mu_g(a) + E \otimes \mu_g(u) \}^* &= \bigoplus_{a \in A_c} (\mu_c(P_c(w)) \otimes \mu_g(w)) \\
&= \bigoplus_{a \in A^*} (\mu_c(P_c(w)) \otimes \mu_g(w))
\end{align*}
\]

**Proof:** It follows from the morphism property, distributivity of multiplication and tensor multiplication, and mixed product property. For instance, in the simple case \( A = \{a, u\} \) with \( A_u = \{u\} \), we have

\[
\begin{align*}
\{ \mu_a(a) \otimes \mu_g(a) + E \otimes \mu_g(u) \}^* &= E \otimes \mu_c(a) \otimes \mu_g(a) + E \otimes \mu_g(u) \\
&\quad \oplus \{(\mu_c(a) \otimes \mu_g(a)) \otimes E \otimes \mu_g(u)\}^2 \oplus \ldots ,
\end{align*}
\]

where

\[
\begin{align*}
\{ \mu_c(a) \otimes \mu_g(a) + E \otimes \mu_g(u) \}^2 &= \mu_c(a) \otimes \mu_c(a) \otimes \mu_g(a) \otimes \mu_g(a) \oplus \\
&\quad \mu_c(a) \otimes E \otimes \mu_g(a) \otimes \mu_g(u) \oplus \mu_c(a) \otimes \mu_c(a) \otimes E \otimes \mu_g(u) \oplus \\
&\quad \mu_c(a) \otimes \mu_g(u) \otimes E \otimes \mu_g(u) \oplus \\
&\quad \mu_c(a) \otimes \mu_g(u) \otimes E \otimes \mu_g(u) \otimes \mu_g(u),
\end{align*}
\]

which is compatible with the claimed formula. Note that morphism and mixed product properties have been repeatedly used. It is easy to see that in general case we have the above formula.

We have the following theorem.

**Theorem 4.4:** The behavior of the parallel composition is the following:

\[
\begin{align*}
l(G_c)(G)(w) &= l_c(P_c(w)) \otimes l_g(w) \\
\text{Proof:} & \quad l(G_c)(G)(w) = \alpha \otimes \mu^* \otimes \beta = (\alpha_c \otimes \alpha_g) \otimes \\
&\bigoplus_{a \in A_c} (\mu_c(a) \otimes \mu_g(a)) \otimes \bigoplus_{a \in A_u} E \otimes \mu_g(a) \}^* \otimes \\
&\quad (\beta_c \otimes \beta_g).
\end{align*}
\]

Using Lemma 4.3

\[
l(G_c)(G) = (\alpha_c \otimes \alpha_g) \otimes \\
\bigoplus_{a \in A^*} (\mu_c(P_c(w)) \otimes \mu_g(w) \otimes (\beta_c \otimes \beta_g) = \\
\bigoplus_{a \in A^*} \{\alpha_c \otimes \alpha_g \otimes (\mu_c(P_c(w)) \otimes \mu_g(w) \otimes (\beta_c \otimes \beta_g)\}.
\]

Finally using the mixed product property we obtain:

\[
l(G_c)(G) = \bigoplus_{a \in A^*} \{[\alpha_c \otimes \mu_c(P_c(w)) \otimes \beta_c] \otimes \mu_g(w) \}^*.
\]

Hence, \( l(G_c)(G)(w) = l_c(P_c(w)) \otimes l_g(w) \) as claimed.

By comparing the definition of Hamadard product with the formula of the last theorem we can view the right hand side as a kind of generalized Hamadard product (in presence of uncontrollable events). We propose the following definition.

**Definition 4.1:** Let \( A = A_c \cup A_u \) with the associated natural projection \( P : A^* \to A_c \). The generalized Hamadard product of two formal power series \( s \) and \( s' \), denoted \( \odot A_u \), is defined by \( (s \odot A_u s')(w) = a(P_c(w)) \otimes s'(w) \).

It follows from theorem 4.4 that

\[
l(G_c)(G)(w) = l_c(P_c(w)) \odot l_g(w).
\]

This can be applied to control of \( \text{max},+ \) automata in a behavioural framework.

Let \( y_{\text{ref}} \) be a specification series, the problem is to find the greatest controller series, denoted \( y_c \) such that \( y_c \odot A_u y \leq y_{\text{ref}} \). Let us introduce the notation

\[
H_y^{A_u} : s \mapsto s \odot A_u y
\]

for the right generalized Hamadard product.

Since \( H_y^{A_u} : \mathbb{R}_{\text{max}}(A) \to \mathbb{R}_{\text{max}}(A) \) is again a residuated mapping (with its residiuated mapping denoted by \( (H_y^{A_u})^\dagger \)), there exists the greatest \( y_c \) such that \( H_y^{A_u}(y_c) \leq y_{\text{ref}} \), namely

\[
y_{\text{ref}} \dagger = (H_y^{A_u})^\dagger (y_{\text{ref}}).
\]

Proposition 3.3 has the following variant in presence of uncontrollable events \( (A_u \neq \emptyset) \).

**Proposition 4.5:** The mapping \( H_y^{A_u} \) is residuated and its residual mapping is given by

\[
(H_y^{A_u})^\dagger(s) = \bigoplus_{w \in A^*} \bigcap_{u \in P_c^{-1}(w) \cap \text{supp}(y)} (s(u) \otimes y(u)) w.
\]

**Proof:** The proof goes along the same lines as that of Proposition 3.3. We obtain: Using (f.1) and (f.2) in dioid \( \mathbb{R}_{\text{max}} \), we show the required inequalities:

\[
(H_y^{A_u})^\dagger(s) = (H_y^{A_u})^\dagger \bigoplus_{w \in A^*} \bigcap_{u \in P_c^{-1}(w) \cap \text{supp}(y)} (s(u) \otimes y(u)) w
\]

where in the last inequality (f.1) has been used and in the second last inequality (ii) of Lemma 4.2, i.e. \( P_c^{-1}(w) \supseteq \{w\} \).

Similarly,

\[
((H_y^{A_u})^\dagger \circ H_y^{A_u})(s) = (H_y^{A_u})^\dagger \bigoplus_{w \in A^*} \bigcap_{u \in P_c^{-1}(w) \cap \text{supp}(y)} (s(u) \otimes y(u)) \otimes y(w) w
\]

The first inequality holds true because from (i) of Lemma 4.2 \( P_c P_c^{-1}(w) = w \) : \( P_c \circ P_c^{-1} \) is identity and we work with series that are nondecreasing on their support, i.e. \( \forall u \in P_c^{-1}(w) \cap \)
supp(y): y(w) ≤ y(u) (because for u ∈ P_w−1(w) we have obviously w ≤ u with respect to the subword order on A∗ defined in Section II). The second inequality follows easily from properties of residuated multiplication (f.2). Hence, $H_y^A$ is residuated according to Definition 2.1 with $(H_y^A)^{-1}$ given by the above formula.

Remark 4.6: We point out the following relationship with the classical supervisory control theory. The residuated mapping $(H_y^A)^{-1}(s)$ plays the role (i.e. is a generalization of) the supremal controllable sublanguage of specification (reference) series s with respect to the plant y and $A_u$. Indeed, $H_y^A(s)$ plays the role of infimal controllable superlanguage of the specification series s with respect to y and $A_u$. It is just an extension to the (max,+) case of the algebraic counterpart of supervisory product defined by coinduction in [14]. Actually, if we denote in the classical supervisory control theory the operator $H_y(K) = \inf C(K, L, A_u)$ the resulting closed-loop system, which corresponds to the infimal controllable superlanguage of the specification language K with respect to plant language L and $A_u$, then it can be shown that this mapping is residuated in the dioid of formal languages and its residuated mapping is nothing else but $H_y(K) = \sup C(K, L, A_u)$. The last proposition can then be viewed as a generalization of the formula for supremal controllable sublanguage of specification (reference) series in [14].

Example 1: We consider a DES (e.g. a manufacturing system) in which three distinct tasks can be done. These tasks, labelled a, b and c, last respectively 3, 4 and 5 units of time. The system can perform the following sequences of tasks: a, ab, abc, abeb, abec, .... This system can be modeled by the (max,+) automaton $G$ displayed in figure 2.(a). The behavior of $G$ can be traduced by the following series in $\mathbb{R}_{max}(A)$:

$$y = 3a(9bc)^*(4b + e).$$

For instance, $y(ab) = 7$ means that the sequence ab will be completed at the earliest at date 7 (considering that the system starts to operate at time 0).

It is assumed that the start of tasks a and c can be delayed (we may decide to postpone the execution of these tasks when they should be performed) or even forbidden (their execution can be prevented). On the contrary, the task b can neither be delayed nor forbidden (this task starts as soon as it can be performed). Denoting $A = \{a, b, c\}$ the set of events (alphabet), we then have $A_s = \{a, c\}$ and $A_u = \{b\}$.

We would like that the system operates at the latest according to the following series:

$$y_{ref} = 4a \oplus 9ab \oplus 14abc.$$ 

This means that the sequences a, ab and abc should be completed at the latest at dates 4, 9 and 14 respectively. In addition, any other sequence of tasks should not happen. This series is recognized by the (max,+) automaton $G_{ref}$ displayed in figure 2.(b).

In order to achieve this goal, we will apply the proposed supervisory control.

First of all, we apply classical results of conventional supervisory control to check that the logical aspects of the specified behavior (in our example to accept the sequences a, ab and abc can be achieved (and restrain this objective if necessary). We then work on automata $G$ and $G_{ref}$ neglecting the durations of events and their associated languages denoted $L(G)$ and $L(G_{ref})$. It is well known that $L(G_{ref})$ is controllable iff $L(G_{ref}) \cap L(G) \subseteq \overline{L(G_{ref})}$, in which denotes the concatenation and $\overline{L(G_{ref})}$ denotes the prefix closure of $L(G_{ref})$. Here, we have $L(G) = \epsilon + a(\epsilon + b)^*(\epsilon + e)$, $L(G_{ref}) = \epsilon + a + ab + abc$ and $L(G_{ref})$ is not controllable since $abc, b \cap L(G) = abc \not\subseteq \overline{L(G_{ref})}$. We then restrict ourselves to the supremal controllable sublanguage of $L(G_{ref})$ given by $L(G_{ref})^\uparrow = \epsilon + a + ab$. And the initial objective $y_{ref}$ is then changed by $y_{ref}^\uparrow = 4a \oplus 8ab$ which is recognized by the (max,+) automaton $G_{ref}^\uparrow$ displayed in figure 2.(c).

Let us notice that in the initial specification the task ab was supposed to be completed before date 9, while the event b can not be delayed, i.e. nothing changes if we require ab to be completed before date 8. Thanks to (10) we can compute $y_{ref}^{opt}$ which is the greatest solution in $\mathbb{R}_{max}(A)$ of $y_C \oplus y_{ref} \leq y$. We obtain $y_{ref}^{opt} = 1a \oplus 1ab$ which is recognized by the (max,+) automaton $G_C$ displayed in figure 2.(d).

Finally, the behavior of the supervised system can be obtained as the parallel composition of $G$ and $G_C$. The obtained (max,+) automaton, denoted $G_s$, is displayed in figure 2.(e).

Let us recall that the behavior of $G_s$ is modelled by $y_C \circ y$. Now, the second approach dealing at the same time with both logical and timing aspects yields according to formula
for \((H^A_y)^a\) \(y_c(a) = \min\{y_{ref}(a) \cdot y(ab), y_{ref}(ab) \cdot y(ab)\} = \min\{4\%3, 8\%7\} = 1\), because \(ab \in supp(y)\) and \(F_c(ab) = b\), i.e. \(\{a \in F^{-1}_c(a \cap supp(y))\} = \{a, ab\}\). Similarly, \(y_c(ab) = y_{ref}(ab) \cdot y(ab) = 8\%7 = 1\), and \(y_c(ab) = \min\{y_{ref}(abc) \cdot y(abc), y_{ref}(abab) \cdot y(abab)\} = \min\{13\%12, 16\%16\} = \varepsilon\). It is easy to check \(y_c(w) = \varepsilon\) for any other \(w \in A^*\). Hence we obtain the same result using directly residuation theory.

V. CONCLUSION

We have presented a control mechanism for (max,+) automata based on the tensor product of their linear representation, i.e. Hamadard product of the corresponding formal power series. Both logical and timing aspects of their control have been studied using behavioral (formal power series) framework. In presence of uncontrollable events we have developed an approach based on a generalized version of Hamadard product and on direct application of residuation theory: both logical and timing aspects of supervisory control are handled at the same time. From a practical point of view, the application of residuation theory requires implementation of efficient algorithms for manipulation with formal power series in several noncommuting variables. In [5] and [4], authors study non-decreasing ones among these series and propose simplification rules. These results should be investigated in future works. Among questions opened for a future investigation one is primordial: is the residuated series of Hamadard product and generalized Hamadard product rational given both plant and specification series are rational?

Moreover it would be nice to handle unobservable events and to develop decentralized and modular control of (max,+) automata. Another work would consist in introducing I/O (max,+) automata in order to leave restrictions on the form of specification series using methods of standard control theory: e.g. precompensator and general feedback control scheme.

REFERENCES