A Max-Plus Algebra of Signals for the Supervisory Control of Real-Time Discrete Event Systems [*]

Guillaume Brat and **Vijay K. Garg**
Department of Electrical and Computer Engineering
The University of Texas at Austin
Austin, TX 78712, USA

March 27, 1998

Abstract

In this paper, we define a max-plus algebra of signals for the evaluation of timing behavior of discrete event systems modeled by timed event graphs. We restrict ourselves to infinite, periodic sequences for which we can compute finite representations called signals. This framework allows us to implement a max-plus algebra for computing supremal controllers for real-time, discrete event systems.

Keywords: max-plus algebra, discrete event systems, real time, supervisory control, periodic signals.

1 Introduction

It has been shown that max-plus algebra can be applied to the modeling of timing behavior in for discrete event systems (DES) [5, 6, 7]. Advances in this field have been extensively reported in [1] and [9]. The same framework has been used by Cofer and Garg to solve the supervisory control problem for real-time DES [3, 4]. Behaviors of DES are captured by Timed Event Graphs (TEG). Certain events are declared controllable and they may be disabled or delayed by a supervisor to restrict the system to a specified behavior. Cofer and Garg defined algorithms to compute supremal controllers for real-time DES. Unfortunately, event sequences are generally infinite, and therefore, some sequences do not have a finite representation. This problem has prevent Cofer and Garg to automate their work.

Our goal is to define a max-plus algebra for event sequences that ultimately exhibit periodic patterns. These sequences, which we call periodic signals, have finite representations. This allows us to automate operations on periodic signals, and therefore, to implement the algorithms of Cofer and Garg [3, 4]. We use periodic signals not only to represent event sequences, but also the delays that can be applied to these sequences.

We first present an overview of max-plus algebras for discrete event systems. Then, we define a finite representation for event sequences, called signals. We then present algorithms for implementing operations on signals. We also define algorithms for computing closures on signals and on matrices of signals. We illustrate our framework with a manufacturing process example. Because of space limitation, we omit proofs that do not offer any difficulty, including the proofs of correctness of our algorithms.

2 Background and Example

Max-plus algebras were studied by Cuninghame-Green [8] in the framework of operation research. Cuninghame-Green observed that non-linear equations can be expressed linearly using an algebra in which sum and product are defined as maximization and conventional addition respectively. In this section, we give a short introduction to the max-plus algebra for discrete event systems. Further details can be found in [3].

Let $\varepsilon = -\infty$, $e = 0$, and Z be the union of the set of integers and $\{\varepsilon\}$. Let $+$ be the conventional addition on Z given that, for any element $a \in Z$, $a + \varepsilon = \varepsilon$. Define $\oplus$ and $\otimes$ such that

- for all $(a, b) \in Z^2$, $a \oplus b = \max(a, b)$
- for all $(a, b) \in Z^2$, $a \otimes b = a + b$

The $\oplus$ operator has the following properties:

- commutativity, i.e., for all $(a, b) \in Z^2$, $a \oplus b = b \oplus a$

[*]supported in part by the NSF Grants ECS-9414780, CCR-9520540, TRW faculty assistantship award, a General Motors Fellowship, and an IBM grant. G. Brat is also employed by Microelectronics and Computer Technology Corporation, Austin, Texas

- associativity, i.e., for all $(a,b,c) \in Z^3$, $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
- idempotency, i.e., for all $a \in Z : a \oplus a = a$
- ε is the identity element, i.e., for all $a \in Z$, $a \oplus \varepsilon = \varepsilon \oplus a = a$

The $\otimes$ operator has the following properties:

- associativity, i.e., for all $(a,b,c) \in Z^3$, $(a \otimes b) \otimes c = a \otimes (b \otimes c)$
- distributivity over $\oplus$, i.e., for all $(a,b,c) \in Z^3$, $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$
- e is the identity element, i.e., for all $a \in Z$, $a \otimes e = e \otimes a = a$
- ε is absorbing with respect to $\otimes$, i.e., for all $a \in Z$, $a \otimes \varepsilon = \varepsilon \otimes a = \varepsilon$

The structure $(Z, \oplus, \otimes)$ is a max-plus algebra, also called a dioid. For example, $1 \oplus 2 = 2$ and $1 \otimes 2 = 3$. This concept can be extended to matrices as follows. If A,$B \in Z^{N\times N}$, then, for all i and j from 1 to N,

$$(A \oplus B)_{ij} = a_{ij} \oplus b_{ij}, \text{ and}$$

$$(A \otimes B)_{ij} = \bigoplus_{k=1}^{N} a_{ik} \otimes b_{kj}.$$

$(Z^{N\times N}, \oplus, \otimes)$ is also a dioid, where the identity elements for $\oplus$ and $\otimes$ are, respectively,

$$\varepsilon = \begin{pmatrix} \varepsilon & \cdots & \varepsilon \\ \vdots & & \vdots \\ \varepsilon & \cdots & \varepsilon \end{pmatrix} \quad \text{and} \quad I = \begin{pmatrix} e & & \varepsilon \\ & \ddots & \\ \varepsilon & & e \end{pmatrix}.$$

In [1, 3], the max-plus algebra has been applied to the modeling of discrete event systems. The behavior of a system is captured by a special class of Petri nets called timed event graphs. Petri nets are widely used as a graphical tool to model distributed processes. A comprehensive review of their definition and use can be found in [10]. A Petri net is a bipartite graph in which vertices are either places or transitions. Edges may go from transitions to places or from places to transitions. Initially, places are marked with zero or more tokens. The presence of tokens in a place indicates that some condition, or state, in the process is satisfied. Transitions are normally associated with events. A transition may be activated, or fired, when all its predecessor places contain at least one token (i.e., some conditions are met that cause an event to occur). Upon firing, one token is removed from each place preceding the transition and one token is added to each of its successors.

In a TEG model, an event sequence corresponds to the successive firing times of a transition. In [1], Baccelli *et al.* describes a method for calculating all event sequences in the system using max-plus algebra. A TEG (with N transitions) is represented by a $N \times N$ delay matrix in the $(Z^{N\times N}, \oplus, \otimes)$ max-plus algebra. The element A_{ij} of a delay matrix A is the time delay associated with the place whose input (output) transition is τ_j (τ_i respectively). The problem can be posed as a set of equations

$$x_i = \bigoplus_{1\le j\le N} A_{ij} x_j \oplus v_i, \text{ where } 1 \le i \le N.$$

x_i is the actual firing time of transition τ_i, and v_i is the earliest time that transition τ_i may fire. This system of equations can be written as

$$x = Ax \oplus v,$$

the least solution of which is $A^* v$ [6] where

$$A^* = \bigoplus_{k\ge 0} A^k.$$

A^*, called the *-delay matrix of A, gives the maximum delay between transitions along infinite paths.

In [3], Cofer and Garg show that max-plus algebra and timed event graphs can be used to analyze the controllability of discrete event systems (DES). Using the max-plus algebra framework, they define conditions under which supremal controllers can be found for any system represented by a TEG. The goal of this paper is to automate their work by defining a max-plus algebra for event sequences that have a finite representation.

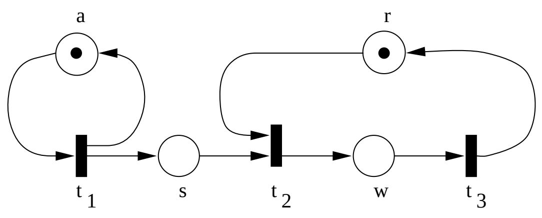

Figure 1: Timed event graph of a manufacturing process.

Example: Manufacturing Process
Consider the timed event graph of Figure 1 which represents a manufacturing process. Upon arrival (represented by transition t_1), parts are first set-up (s) in a machine queue, and then worked (w) in order of arrival. Transition t_2 represents a part leaving the queue, and transition t_3 the completion of a part. Each operation takes a constant amount of time. However, the inter-arrival time (a) may vary due to the work floor schedule. Moreover, the machine reset time (r) is not constant either. We, however, assume that both a and r ultimately follow periodic patterns. The delay functions are defined as follows:

$$s(x) = x + 1$$

$$w(x) = x+4$$

$$a(x(k)) = \begin{cases} x(k)+5 & \text{if } k \text{ is odd} \\ x(k)+7 & \text{if } k \text{ is even} \end{cases}$$

$$r(x(k)) = \begin{cases} x(k)+4 & \text{if } k \bmod 5 = 0 \\ x(k)+1 & \text{otherwise} \end{cases}$$

where k is the k^{th} occurrence of the event type x.

3 Definition of Signals

We use sequences of natural numbers to model timing behavior of DES. Each natural number represents the date at which events, indexed in the set of natural numbers, of the same type occur. A sequence X is defined as a function from the set of natural numbers, called indices, to the union of the set of natural numbers and $\{\varepsilon\}$ such that

$$\forall k \geq 0 : (X[k] = \varepsilon) \Rightarrow (\forall j < k : X[j] = \varepsilon).$$

A sequence X is periodic if and only if there exist k,C, and n such that. for all $i \geq n$, $X[i+k] = X[i] + C$. For example, the sequence $\{0, 4, 7, 9, 11, 13, \ldots\}$ represents event occurrences that happen at time zero, four, seven, and every two time units from then on. This sequence is infinite, but it can be represented by an initial finite sequence $\{0, 4, 7\}$, and an infinite periodic sequence $\{9, 11, 13, \ldots\}$.

By definition, not all sequences can be modeled using a finite number of bits. Therefore, we restrict ourselves to periodic sequences. We view a periodic sequence as an initial finite sequence (called transitory sequence) and a periodic finite sequence (called period). Periodic sequences have a finite representation that we call periodic *signals*.

Definition 1 *Let I be the set of integers. A signal is defined as a tuple $(T; P)$ where*
1) $T = (t_1, t_2, \ldots, t_n)$ is a finite list of $n \geq 0$ elements (called transitory steps) in $I \cup \{\varepsilon\}$ s.t.

$$\forall 0 \leq k \leq n : (t_k = \varepsilon) \Rightarrow (\forall j < k : t_j = \varepsilon).$$

2) $P = (p_1, p_2, \ldots, p_m)$ is a finite list of $m \geq 0$ integers (called periodic steps) s.t. $\sum_{k=1}^{k=m} p_k \geq 0$.

Each t_i and p_i represents the time elapsed between occurrences of consecutive indices of a same type of events. For example, the sequence $X = \{0, 4, 7, 9, 11, 13, \ldots\}$, can be represented by the signal $x = ((0, 4, 3); (2))$. In fact, all periodic sequences can be represented by a periodic signal and vice versa.

There are many representations of a same signal. For example, $x = ((2, 3); (1, 3, 1, 3))$ and $y = ((2); (3, 1))$ represent the same signal. For every signal, there exists a canonical form that can be computed by eliminating any suffix of the transitory sequence matching a suffix of the period, and then, finding the minimal representation of the period. In our example, y is the canonical form of x. Canonizing a signal can be performed using a $O(m+n)$ algorithm. The canonical form gives us a simple means to compare two signals. Conversely, it is sometime convenient (as in our algorithms) to expand the representation of two signals so that their transitory sequences (periods) have the same number of transitory (periodic respectively) steps. This operations is called homogenizing two signals and can be performed using an $O(m^2+n)$ algorithm.

We now define some notations that will be used in the next sections. The function $T(x)$ maps a signal $x = (T; P)$ to its transitory sequence T. Similarly, the function $P(x)$ maps a signal $x = (T; P)$ to its period P. Finally, the slope $\sigma(x)$ of a periodic signal $x = ((t_1, \ldots, t_n); (p_1, \ldots, p_m))$ is defined as $\sigma(x) = \frac{1}{m}\sum_{i=1}^{m} p_i$. For example, if $x = ((0, 4, 7); (2))$, then $T(x) = (0, 4, 7)$, $P(x) = (2)$, and $\sigma(x) = 2$. Note that we limit our work to signals with non-negative slopes. Finally, $\mathcal{Z}$ is defined as the union of $\{\varepsilon, \varepsilon, \ldots\}$ and the set of periodic signals.

Note, that, in [9], Gaubert defines a similar class of signals, which he represents using a polynomial form. The difference between his work and ours is covered in subsequent sections.

4 Operations on Signals

4.1 Max operation

Definition 2 *Given two signals x and y, and their underlying sequences X and Y, the signal $w = \max(x, y)$, also denoted $w = x \oplus y$, is defined by its underlying sequence $W = \max(X, Y)$, which is defined as the pointwise maximization of the sequences X and Y.*

We also present an $O(m^2 + n + N)$ algorithm (where $n = |T(x)| \oplus |T(y)|$, $m = |P(x)| \oplus |P(y)|$ and N is defined in Step 3 of the algorithm) to compute the max of two signals (x and y respectively).

```
maximize(x,y):
1. If $\sigma(x) = \sigma(y)$ then $T(x \oplus y) = T(x) \oplus T(y)$,
                                  and $P(x \oplus y) = P(x) \oplus P(y)$;
2. elsif $\sigma(x) < \sigma(y)$ then maximize(y,x);
3. else /* let $S(i,x) = \sum_{k=1}^{i} p_k, p_i \in P(x)$ */
     $x_l = \min\{S(i,x) - i\sigma(x), 1 \leq i \leq m\}$;
     $y_u = \max\{S(i,y) - i\sigma(y), 1 \leq i \leq m\}$;
     $N = (x_l - X[0] + y_u + Y[0])/(\sigma(x) - \sigma(y))$;
     $X[-1] = 0$; $Y[-1] = 0$;
     for i from 1 to $N + n$ do
         $T(x \oplus y)[i] = ((X[i] \oplus Y[i])$
                              $-(X[i-1] \oplus Y[i-1])$;
     $P(x \oplus y) = P(x)$;
4. Canonize($x \oplus y$);
```

The maximization of an infinite number of signals does not necessarily result in a periodic signal. The underlying sequence of the signal $y = ((0);(1))$ is $Y = \{0,1,2,3,4,\ldots\}$. The set $\mathcal{X} = \{x_k, 0 \leq k : x_k = ((\sum_{i=0}^{j} Y[i], 0 \leq j \leq k);(0))\}$ is an infinite set of periodic signals; but $\bigoplus_{x \in \mathcal{X}} x$ corresponds to the sequence $\{0,1,3,6,10,15,21,\ldots\}$, which is not a periodic sequence.

4.2 Backshift operator

Initial marking in TEGs are modeled using the backshift operator. Tokens initially present in a given place immediately contribute to the enabling of its output transition, even if the place's delay is not null.

Definition 3 *Let $x = (T,P)$ be a periodic signal in which $T = (t_1, t_2, \ldots, t_n)$. Then, the backshift operator γ is defined as $\gamma x = \gamma(x) = ((\varepsilon, t_1, t_2, \ldots, t_n), P)$.*

In our manufacturing process example, the place with the delay function a initially contains a token. Therefore, the actual delay associated with this place is $a\gamma$. Let x_i denote the occurrence times of event t_i. Then, the manufacturing process is described by the following equation:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} a\gamma & \varepsilon & \varepsilon \\ s & \varepsilon & r\gamma \\ \varepsilon & w & \varepsilon \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \oplus \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$$

where $v_1 = v_2 = v_3 = (();(0)$.

Note that $\mathcal{Z}$ is closed under the backshift operator γ^i, for all $i \geq 1$. The backshift operator also distributes over the max operator.

4.3 Periodic delay function

This section defines and characterizes a function that can delay a periodic signal according to another periodic signal.

Definition 4 - Delay -
Let $x = ((t_1,\ldots,t_n);(p_1,\ldots,p_m))$ be a periodic signal, and δ_d be a periodic delay function such that $d = ((t'_1,\ldots,t'_n);(p'_1,\ldots,p'_m))$.
Let t_i (t'_j) be the first element of $T(x)$ ($T(d)$ respectively) not equal to ε.
Then,

$$P(\delta_d(x)) = (p_1 + p'_1, \ldots, p_m + p'_m)$$

and, if $j \geq i$, $T(\delta_d(x))$ is equal to

$$(t_1, \ldots, t_{j-1}, t_j + t'_j, \ldots, t_n + t'_n)$$

otherwise,
$(t_1, \ldots, t_{i-1}, t_i + \sum_{l=j}^{i} t'_l, t_{i+1} + t'_{i+1}, \ldots, t_n + t'_n)$.

The complexity of the algorithm implementing the delay operator is $O(m^2 + n)$ where $(m = |P(x)| \oplus |P(d)|$ and $n = |T(x)| \oplus |T(d)|)$ because of the need to homogenize the input and delay signals.

```
delay(x,d): y
1. for i from 1 to n do
     if ($T(x)[i] = \varepsilon$) then
         $T(y)[i] := \varepsilon$;
         if ($T(d)[i] \neq \varepsilon$) then init := init + $T(d)[i]$;
     elsif ($T(d)[i] = \varepsilon$) then $T(y)[i] := T(x)[i]$;
     else $T(y)[i] := T(x)[i] + T(da)[i]$ + init;
         init := 0;
2. P(y) := pointwise addition of P(x) and P(d);
3. Canonize (y);
```

Note that $\mathcal{Z}$ is closed under the operation defined by a periodic delay, and that the composition of two periodic delay functions is commutative, associative, and the neutral element is the zero delay function. However, the composition of a periodic delay function and the backshift operator is, in general, not commutative except when the delay is constant. In fact, if d and x are arbitrary periodic signals, $\gamma\delta_d(x) = \delta_{\gamma d}(\gamma x)$.

In our manufacturing process example, our delay functions can be defined in terms of periodic signals. The constant delays s and w can be expressed with $((1);())$ and $((4);())$ respectively. The periodic delays a and r can be represented by $((7);(-2;2))$ and $((); (4,-3,0,0,0))$.

Our periodic delay function should not be confused with the $\otimes$ operator in [1, 9], which is defined as the sup-convolution of two sequences. The effect of the sup-convolution of an input signal and a delay signal are not easily predictable. For example, delaying odd-indexed events of a signal by seven and even-indexed events by five (as in the a delay function in our manufacturing process example), cannot be expressed with the $\otimes$ operator.

5 Closure Operations

The least solution of $x = Ax \oplus v$ is A^*v, where $A^* = \bigoplus_{k \geq 0} A^k$ [1]. The next two sections describe algorithms to compute A^*.

5.1 Periodic *-delay function

Definition 5 *A periodic *-delay function $(d\gamma)^*$ is defined by a periodic delay signal d,where the slope of d is null, as $(d\gamma)^* = \bigoplus_{k \geq 0} (\delta_d \gamma)^k$.*

To show the completeness of the *-delay function in $\mathcal{Z}$, we need to prove that $\bigoplus_{k \geq 0} (\delta_d \gamma)^k$ is equal to a finite maximization. In the rest of this section, we assume that the periods of the periodic delay signal and the input periodic signals are homogeneous and contain m steps. We also define the maximum index of the transitory pattern of the input signal to be n. Due to space limitation, we only outline the major steps of the proof.

The first step of the proof consists of showing that, for any signal x, $\bigoplus_{k\geq 0}(\delta_d\gamma)^k(x)$ can be re-written as the maximization over an infinite set of signals that are rooted at time instants corresponding to the elements of X (the underlying sequence of x). Thus, we want to show that

$$\bigoplus_{k\geq 0}(\delta_d\gamma)^k(x) = \bigoplus_{k\geq 0}\gamma^k x_k,$$

where $x_k = \gamma^k((X[k]);(D[k+1],\ldots,D[k+m]))$ and D is the underlying sequence of the delay signal d. For example, if $x = ((0);(2))$ and $d = (();(1,-1))$, then $\bigoplus_{k\geq 0}(\delta_d\gamma)^k(x)$ corresponds to the maximization of the signals corresponding to the following sequences:

$$\begin{array}{ll} & \{0,1,3,5,7,9,11,\ldots\} \\ \oplus & \{\varepsilon,0,2,3,6,7,10,\ldots\} \\ \oplus & \{\varepsilon,\varepsilon,1,2,4,6,8,\ldots\} \\ \oplus & \{\varepsilon,\varepsilon,\varepsilon,1,3,4,7,\ldots\} \\ \oplus & \ldots \end{array}$$

which can be re-written as

$$\begin{array}{ll} & \{0,0,1,1,2,2,3,\ldots\} \\ \oplus & \{\varepsilon,1,2,2,3,3,4,\ldots\} \\ \oplus & \{\varepsilon,\varepsilon,3,3,4,4,5,\ldots\} \\ \oplus & \{\varepsilon,\varepsilon,\varepsilon,5,6,6,7,\ldots\} \\ \oplus & \ldots \end{array}$$

The second part of the proof is simply a matter of comparing the slopes of the x_k's to the slope of x. Observe that the x_k's have the same slope $(1/m\sum_{i=1}^{m} D[k+i])$. Using that fact, it is quite easy to show that, in any case, $\bigoplus_{k\geq 0}\gamma^k x_k$ can be computed using, at most, the first $(n+2m)$ x_k signals. Therefore, it shows that $(d\gamma)^*(x)$ is the maximization of a finite number of signals. We now give an algorithm to compute such quantity.

```
*-delay (x,d): y
1. Compute $\sigma(x_i) = 1/m\sum_{i=n+1}^{m} D[i]$;
2. if ($\sigma(x_i) \geq \sigma(x)$) then
       y := $\bigoplus_{k=0}^{n+m} x_k$;
3. else T(z) := T($\bigoplus_{k=n+1}^{n+m} x_k$);
       P(z) := P($\bigoplus_{k=n+m+1}^{n+2m} x_k$);
       y := Maximize(z,$\bigoplus_{k=0}^{n} x_k$);
4. Canonize (y);
```

The complexity of the algorithm lies in Steps 2 and 3, each of which is in $O((n+m)(m^2+n+N))$ where m^2+n+N represents the cost of maximizing two signals. Therefore, the complexity of the **-delay* function is $O((n+m)(m^2+n+N))$.

5.2 Closure matrices of signals

Our algorithm to compute the *-delay matrix of a matrix representing a TEG is based on the algorithm developed in [9] and described by the following formula:

$$A^*_{ij} = \bigoplus_{p\in\mathcal{P}_{ij}} p\otimes(e\oplus\bigoplus_{C\in\mathcal{A}(p)}\bigotimes_{c\in C} c^+). \quad (1)$$

A is a matrix representing a system and $\mathcal{G}(A)$ is the graph structure of the Petri net associated with matrix A. $\mathcal{P}_{ij}$ is the set of elementary paths from j to i in $\mathcal{G}(A)$, and $\mathcal{A}(p)$ is the set of elementary circuits accessible from a path p. A circuit C is said to be accessible from a path p if $C\cup\{p\}$ is a connected subgraph of $\mathcal{G}(A)$.

We cannot directly apply Gaubert's formula because it assumes that $\otimes$ is commutative, and, in our max-plus algebra, the delay function (which corresponds to $\otimes$) is not commutative. We have to alter Gaubert's formula by taking into account the position of the elementary circuits in the considered elementary paths. Thus, after rearranging terms, Gaubert's formula becomes

$$A^*_{ij} = \bigoplus_{p\in\mathcal{P}_{ij},C\in\mathcal{A}(p)}(c^*_i a_{ik}c^*_k a_{kl}c^*_l\ldots c^*_m a_{mj}c^*_j)(2)$$

where each elementary path $p\in\mathcal{P}_{ij}$ can be written as $a_{ik}a_{kl}\ldots a_{mj}$ and the $c^*_i, c^*_k, c^*_l, \ldots c^*_m, c^*_j$ are the transitive closures of the delays in elementary circuits containing transitions $\tau_i,\tau_k,\tau_l,\ldots\tau_m,\tau_j$.

Let us illustrate our formula by computing A^*_{31} in our manufacturing process example. This element of A^* corresponds to the closure of all paths from t_1 to t_3. There is only one elementary path from t_1 to t_3, which means that $\mathcal{P}_{31}=\{\omega s\}$. Along this path, there are two distinct elementary circuits, which gives us $\mathcal{A}(\omega s)=\{\{wr\gamma\},\{a\gamma\},\{wr\gamma,a\gamma\}\}$. Therefore,

$$\begin{aligned} A^*_{31} &= (wr\gamma)^*ws\oplus ws(a\gamma)^*\oplus(wr\gamma)^*ws(a\gamma)^* \\ &= (wr\gamma)^*ws(a\gamma)^* \end{aligned}$$

Using the same technique, we can calculate each element of A^*. This leads to

$$A^* = \begin{pmatrix} (a\gamma)^* & \varepsilon & \varepsilon \\ (r\gamma w)^*s(a\gamma)^* & (r\gamma w)^* & (r\gamma w)^*r\gamma \\ (wr\gamma)^*ws(a\gamma)^* & (wr\gamma)^*w & (wr\gamma)^* \end{pmatrix},$$

which corresponds to the matrix computing by Cofer and Garg.

Moreover, it yields

$$\begin{aligned} x_3 &= (wr\gamma)^* ws(a\gamma)^* [\boxed{0}] \\ &= (wr\gamma)^* ws[((0);(5,7))] \\ &= (wr\gamma)^* [((5);(5,7))] \\ &= [((5,5,7);(5,7,8,5,5,6,5,8,5,6))] \end{aligned}$$

which corresponds to the following infinite periodic sequence:

$$x_3 = \{5, 10, 17, 22, 29, 37, 42, 47, 53, 58, 66, 71, 77, \ldots\}$$

Even though delay matrices can be rather sparse, *-delay matrices are usually very dense. In [2], we define composition/decomposition techniques that do not require the construction of entire *-delay matrices. The first composition technique, called sequential composition, leads to the decomposition of a graph based on its strongly connected components (SCCs). The second composition technique, called synchronization, divides a graph along its elementary circuits.

6 Conclusion

In this paper, we have defined a composable max-plus algebra of periodic signals that can represent the timing of events in real-time discrete event systems. A signal is a finite representation of infinite event sequences. We defined and analyzed the complexity of several algorithms that implement synchronization and delay functions on signals as well as the closures of transition delay matrices. Our algebra constitutes an implementation of the max-plus algebra of Cofer and Garg [3, 4], which can compute supremal controllers for real-time discrete event systems.

References

[1] F. Baccelli, G. Cohen, G.J. Olsder, and J.-P. Quadrat. *Synchronization and Linearity: an Algebra for Discrete Event Systems.* John Wiley and Sons, 1992.

[2] G. Brat and V. K. Garg. Composability of the max-plus algebra of signals for the evaluation of real-time software systems. Submitted to the 1998 Workshop on Formal Methods in Software Practice, 1997.

[3] D. D. Cofer and V. K. Garg. A max-algebra solution to the supervisory control problem for real-time discrete event systems. In G. Cohen and J.-P. Quadrat, editors, *Lecture Notes in Control and Information Sciences 199: 11th International Conference on Analysis and Optimization of Systems*, 1994.

[4] D. D. Cofer and V. K. Garg. Supervisory control of real-time discrete event systems using lattice theory. In *Proceedings of the 33rd IEEE Conference on Decision and Control*, pages 978–983, December 1994.

[5] G. Cohen, D. Dubois, J.-P. Quadrat, and M. Viot. A linear system-theoretic view of discrete event processes and its use for performance evaluation in manufacturing. *IEEE Transactions on Automatic Control*, AC-30:210–220, 1985.

[6] G. Cohen, P. Moller, J.-P. Quadrat, and M. Viot. Dating and counting events in discrete event systems. In *Proceedings of the 25th IEEE Conference on Decision and Control*, pages 988–993, 1986.

[7] G. Cohen, P. Moller, J.-P. Quadrat, and M. Viot. Algebraic tools for the performance evaluation of DES. *Proceedings of the IEEE*, 77(1):39–58, 1989.

[8] R. A. Cuninghame-Green. *Minimax Algebra.* Springer Verlag, 1979. Number 166 in Lecture Notes in Economics and Mathematical Systems.

[9] S. Gaubert. *Théorie Linéaire des Systèmes dans les Dioïdes.* PhD thesis, Ecole des Mines de Paris, Paris, 1992.

[10] T. Murata. Petri nets: Properties, analysis, and applications. *Proceedings of the IEEE*, 77(1):541–580, 1989.