A (Max,+) Algebra for Periodic Timed Discrete Event Systems

Guillaume Brat and Vijay K. Garg

TR-PDS-1998-001 January 1998

Parallel & Distributed Systems group
Department of Electrical & Computer Engineering
University of Texas at Austin
Austin, Texas 78712

A (Max,+) Algebra for Periodic Timed Discrete Event Systems

Guillaume Brat* Vijay K. Garg†

Parallel and Distributed Systems Laboratory
email: pdslab@ece.utexas.edu
Electrical and Computer Engineering Department
The University of Texas at Austin,
Austin, TX 78712

January 14, 1998

Abstract

In this paper, we define a (max,+) algebra of signals for the evaluation of timing behavior of discrete event systems modeled by timed event graphs. We restrict our work to infinite, periodic sequences for which we can compute finite representations called signals. This paper describes and analyzes the algorithms used to implement our algebra. This effort resulted in the creation of an automated tool that can compute supremal controllers for timed discrete event systems. Our work differs from Gaubert's (max,+) algebra [9] in the definition of the delay operation. We show that our pointwise delay is more expressive than Gaubert's delay, which is defined as the sup-convolution of two signals.

1 Introduction

More recently, (max,+) algebra has been applied to the study of DES [5, 6, 7]. Using (max,+) algebra, DES can be described as a set of linear equations. A comprehensive study of the problem has been reported in [1] and [9]. Building on these results, Cofer and Garg showed that (max,+) algebras can also be applied to solve supervisory control problems for real-time DES [3, 4]. They defined algorithms to compute supremal controllers for DES. As Baccelli and Gaubert, they see DES as finite sets of events which occur infinitely often in a discrete time space. Thus, each event is completely defined by an infinite sequence of time occurrences. In general, infinite sequences of natural numbers do not have a finite representation. Therefore, the work of Cofer and Garg has not been automated.

The goal of our work is to define a (max,+) algebra that works on a finite representation of infinite sequences of events which can be used to automate Cofer and Garg's control algorithms.

*supported 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

†supported in part by the NSF Grants ECS-9414780, CCR-9520540, TRW faculty assistantship award, a General Motors Fellowship, and an IBM grant.

Therefore, in this paper, we define an event representation called a periodic signal. A signal is a finite representation of an infinite, periodic sequence of time values representing instances of a same event. We define a (max,+) algebra on periodic signals, including operators to model synchronization and delay operations. We also present and prove the correctness of algorithms that implement these operators.

In a related work, Gaubert also defines a (max,+) algebra of signals [9], where signals follow periodic patterns. The complexity of the algorithms implementing Gaubert's algebra is essentially similar to the complexity of our algorithms. Our work differs from Gaubert's work in the definition of the delay operator. Gaubert defines a delay as the sup-convolution of two signals while we define it as a pointwise delay of the sequences underlying two signals. In section 4.4, we show that our (max,+) algebra is more expressive since Gaubert's delay operator can be modeled using our delay operator while the converse is not true.

The second section gives an overview of (max,+) algebra. The third section presents the definition of periodic signals and describes algorithms to modify their finite representations. The fourth section defines basic operators to manipulate signals. The fifth section describes algorithms to perform closure operations on expressions made of signals or on matrices of the same type of expressions. The sixth section explains how the (max,+) algebra of signals can perform Cofer and Garg's algorithms to test controllability and compute supremal controllers. Then, the seventh section shows the implementation of our work, which resulted in a C++ tool. Finally, the last section presents our conclusions and future work.

2 Background on (max,+) Algebra

Cuninghame-Green studied (max,+) algebras in the framework of operation research [8]. 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,+) algebra for discrete event systems. Further details can be found in [1, 2, 3, 9].

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,

- $\forall a \in Z,\ a + \varepsilon = \varepsilon + a = \varepsilon$

Define two binary operations in Z, $\oplus$ and $\otimes$, such that

- $\forall a, b \in Z,\ a \oplus b = \max(a, b)$
- $\forall a, b \in Z,\ a \otimes b = a + b$

The $\oplus$ operator has the following properties:

- commutativity: $\forall a, b \in Z,\ a \oplus b = b \oplus a$
- associativity: $\forall a, b, c \in Z,\ (a \oplus b) \oplus c = a \oplus (b \oplus c)$
- idempotency: $\forall a \in Z : a \oplus a = a$

- ε is the identity element: $\forall a \in Z,\ a \oplus \varepsilon = \varepsilon \oplus a = a$

The $\otimes$ operator has the following properties:

- associativity: $\forall a, b, c \in Z,\ (a \otimes b) \otimes c = a \otimes (b \otimes c)$
- distributivity over $\oplus$: $\forall a, b, c \in Z,\ a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$
- e is the identity element: $\forall a \in Z,\ a \otimes e = e \otimes a = a$
- ε is absorbing with respect to $\otimes$: $\forall a \in Z,\ a \otimes \varepsilon = \varepsilon \otimes a = \varepsilon$

The structure $(Z, \oplus, \otimes)$ forms a dioid called the (max,+) algebra. This algebra is one of many exotic algebras which have been applied to optimal control problems, asymptotic analysis in statistical physics, and decision problems [10]. Because of the definition of $\oplus$, there is a natural order associated with the (max,+) algebra

- $\forall a, b \in Z,\ a \leq b \Leftrightarrow a \oplus b = b$

The (max,+) algebra on Z can be extended to matrices as follows. Let $1 \leq i, j \leq m$. Then,

- $\forall A, B \in Z^{m \times n},\ (A \oplus B)_{ij} = a_{ij} \oplus b_{ij}$
- $\forall A \in Z^{m \times p}$ and $B \in Z^{p \times n}$, then $(A \otimes B)_{ij} = \bigoplus_{k=1}^{p} a_{ik} \otimes b_{kj}$

$(Z^{N \times N}, \oplus, \otimes)$ 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,+) 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 (TEG). 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 [12]. 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 a system. A TEG (with N transitions) is represented by a $N \times N$ delay matrix in $(Z^{N \times N}, \oplus, \otimes)$. 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

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

where x_i is the actual firing time of transition τ_i, and v_i is the earliest time at which 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

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

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

In [2], Cofer and Garg apply (max,+) algebra and event graphs to the controllability of discrete event systems. In their model, the behavior of a system is characterized by sequences of events. These sequences are generally infinite, and their computation cannot be automated. In this paper, we define a (max,+) algebra on periodic signals (with a finite representation) that can be applied to the same type of problems. In the rest of the paper, we use their manufacturing process example to our algebra.

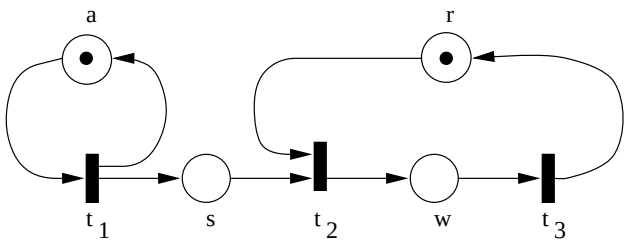

Figure 1: Timed event graph of a manufacturing process.

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:

$$\begin{aligned}
s(x) &= x + 1 \\
w(x) &= x + 4 \\
a(x(k)) &= \begin{cases} x(k) + 5 & \text{if } k \text{ is odd} \\ x(k) + 7 & \text{if } k \text{ is even} \end{cases} \\
r(x(k)) &= \begin{cases} x(k) + 4 & \text{if } k \bmod 5 = 0 \\ x(k) + 1 & \text{otherwise} \end{cases}
\end{aligned}$$

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

3 Periodic Signals

This section defines the concept of periodic signals, describe their basic properties, and presents algorithms to compute the canonical form of a periodic signal and to homogenize periodic signals.

3.1 Definition

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 non-null 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

$$\forall k \geq 1 : (X[k] = \varepsilon) \Rightarrow (\forall j < k : X[j] = \varepsilon).$$

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 $\{4, 7, 9, 11, 13, \ldots\}$ represents event occurrences at time four, seven, and every two time units from then on. Although, this sequence is infinite, it can be represented by an initial finite sequence $\{4, 7\}$ 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, where a periodic sequence consists of an initial finite sequence (called transitory sequence) followed by 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 $I \cup \{\varepsilon\}$ such that

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

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

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 $X = \{4, 7, 9, 11, 13, \ldots\}$, can be represented by the signal $x = ((4, 3); (2))$. In fact, all periodic sequences can be represented by a periodic signal and vice versa. The sequence X associated to a signal x is defined as follows. Let L be the list $L = T.P^{\omega}$ (using traditional notations from automata theory). If P is empty, then the list $\{0\}$ is used instead. X is defined as follows:

$$X[0] = 0, \text{ and } X[i] = X[i-1] + L[i] \text{ for } i > 1.$$

The representation of a sequence as a periodic signal is quite concise. Moreover, given any index k, the value $x[k]$ of a signal x can be calculated using at most $O(m+n)$ operations.

$$x[k] = \begin{cases} \sum_{i=1}^{k} c_i & \text{if } k \leq n \\ \sum_{i=1}^{n} c_i + q \times \sum_{i=1}^{m} p_i + \sum_{i=n+qm}^{k} p_i, \text{ where } q = \lfloor (k-n)/m \rfloor & \text{otherwise} \end{cases}$$

We now introduce some notations that facilitate the understanding of this paper.

Definition 2 - Notations -

- *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.*
- *The slope of a periodic signal $x = ((t_1, \ldots, t_n); (p_1, \ldots, p_m))$ is defined as $\sigma(x) = \frac{1}{m} \sum_{i=1}^{m} p_i$.*
- *$|L|$ is a function returning the length of a list L of (periodic or transitory) steps*

For example, if $x = ((4, 7); (2))$, then $T(x) = (4, 7)$, $P(x) = (2)$, $|T(x)| = 2$, $|P(x)| = 1$, and $\sigma(x) = 2$. Note that, in this work, signals have non-negative slopes since the sum of the elements of the period must be positive or null.

Definition 3 - Sets of periodic signals -

1. *Let $\mathcal{Z}$ be the union of the set of periodic signals as in Definition 1 and $\{\varepsilon, \varepsilon, \ldots\}$.*
2. *Let $\mathcal{N} \subset \mathcal{Z}$ be the set of periodic signals where non-ε elements of the transitory sequence and elements of the periods are natural numbers; these signals are called monotonic signals.*

As in the traditional (max,+) algebra described in the previous section, we can define a natural ordering on periodic signals. It is easy to verify that the following relation defines only a partial order in $\mathcal{Z}$.

Definition 4 - Signal ordering -
For any given signals x and y in $\mathcal{Z}$ and their respective underlying sequences X and Y,

$$X \leq Y \Leftrightarrow \forall i \geq 1 : X[i] \leq Y[i].$$

The understanding of algorithms presented in subsequent sections is facilitated by the following two-dimensional representation of periodic signals. The horizontal axis represents event indices, while the vertical axis represents time. Figure 2 illustrates the representations of two signals $x = ((4, 7); (2))$ and $y = ((4); (2, -2))$. The time values for x's events are represented with $+$'s, and the time values for y's events are represented with $\times$'s. Observe that x is a monotonic signal, and hence, its events occur, in time, in the order of their indices. Signal y, on the other hand, is not monotonic since all the odd-indexed events occur at time four and all even-indexed events occur at time six.

3.2 Canonical Representation

Consider the signals $x = ((2, 3); (1, 3))$ and $y = ((2); (3, 1))$. It is easy to see that sequences corresponding to these signals are identical. This leads to the following definition.

Definition 5 - Equivalence -
The signals $x = (T; P)$ and $y = (U; Q)$ are equivalent (denoted by $x \sim y$) if their sequences X and Y are equal.

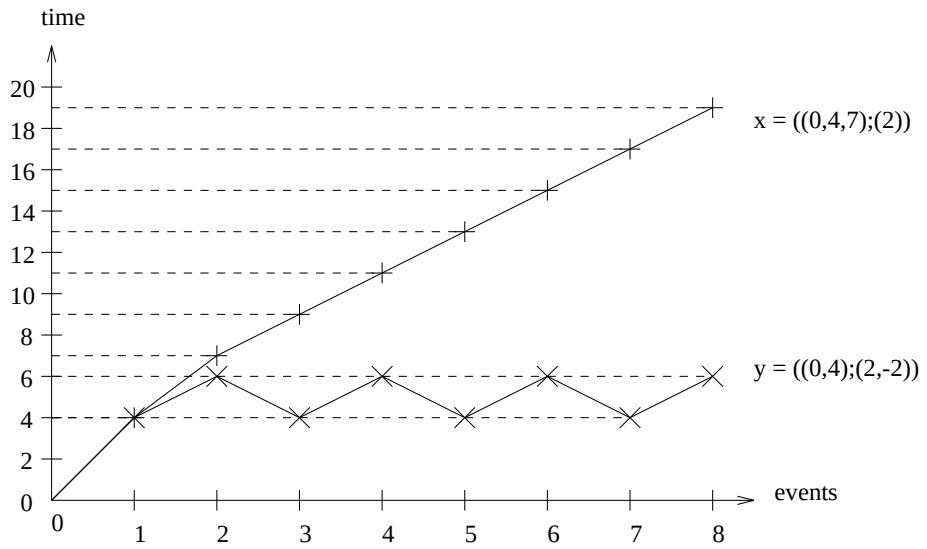

Figure 2: Space representation of periodic signals.

The following lemma gives equivalences of different signals. We use the traditional notation of automata theory, i.e., $x.y$ is the list obtained by concatenating list x and list y, and x^i is the list obtained by concatenating i lists equal to x.

Lemma 1 *The following statements characterize signal equivalence:*

1. $(T;P) \sim (T;P^j)$ *for any* $j > 0$.
2. $(T.D;P.D) \sim (T;D.P)$

Proof: Part 1. Observe that the list P^ω is identical to $(P^j)^\omega$.
Part 2. Again, observe that the list L constructed for both signals is identical.

■

Since there are many signal representations of a same sequence, it is natural to consider whether there is a canonical signal for a sequence. This question is addressed by the following definition.

Definition 6 - Canonicity -
A signal $x = (T;P)$ *is called canonical if the following conditions are true.*

A.1 *There does not exist any positive natural* j *such that the last* j *numbers of* T *match the last* j *numbers of* P.

A.2 *There does not exist a list* Q *and a number* $i > 1$ *such that* $P = Q^i$.

In our example, signals $x = ((2,3);(1,3))$ and $y = ((2);(3,1))$ are equivalent, but x is not canonical while y is canonical. We now show that there cannot exist two different canonical signals for a sequence.

Theorem 1 *If x and y are canonical, and they are equivalent, then x equals y.*

Proof: Assume that x and y are two canonical signals that are also equivalent, i.e., their sequences (X and Y respectively) are equal. Assume also that x and y are different, i.e., either $T(x) \neq T(y)$, or $P(x) \neq P(y)$ or both.

- $T(x) \neq T(y)$:
  Assume if possible, that one is not a prefix of the other. This implies that there exists i such that $T(x)[i] \neq T(y)[i]$. However, this implies that the sequences for x and y are different, which violates the equivalence assumption.
  So, now assume that $T(x)$ is a proper prefix of $T(y)$. Using the traditional notation of automata theory (i.e., $u.v$ is the concatenation of u and v), we prove that y violates Property A.1 in Definition 6.

$$
\begin{aligned}
\exists z \text{ non null} : T(y) = T(x).z \quad &\Rightarrow \quad \exists w : P(x) = z.w && (x \text{ is canonical}) \\
&\Rightarrow \quad \exists w : P(x)^{|P(y)|+1} = z.(w.z)^{|P(y)|}.w \\
&\Rightarrow \quad \exists w : z.P(y)^{|P(x)|}.w = z.(w.z)^{|P(y)|}.w && (x \text{ and } y \text{ are equivalent}) \\
&\Rightarrow \quad \exists w, v : z.P(y)^{|P(x)|}.w = z.(v.z)^{|P(x)|}.w \\
&\Rightarrow \quad \exists v : P(y) = v.z
\end{aligned}
$$

  Thus, according to Property A.1 of Definition 6, y is not canonical.

- $T(x) = T(y)$:
  Again, if $P(x)$ and $P(y)$ are not identical, and one of them is not a prefix of the other, we get a contradiction to the equivalence of x and y.
  If one is a prefix of the other, from the equivalence of x and y, we can conclude that the signal with the longer prefix is not canonical.

■

Using the notation of Definition 1, the following function will convert a signal to its canonical form. The following theorem proves its correctness.

canonize(signal x)
{
  1. *if* ($c_n = p_m$) *then*
     T(x) := $(c_1, \ldots, c_{n-1})$;
     P(x) := $(c_n, p_1 \ldots, p_{m-1})$;
     Update values for n, p_i*'s, and* c_i*'s;*
     re-iterate Step 1;
  2. *Find largest* i *and* Q *such that* $Q^i = P$;
     P(x) := Q;
}

Theorem 2 *The canonize function returns a canonical signal that is equivalent to its input.*

Proof: From Lemma 1, each step in the *canonize* function preserves the equivalence of signals. It remains to be shown that neither A1 nor A2, from the definition of canonicity, is applicable to the signal obtained after the second step. Clearly, A2 is not applicable since we choose the largest i such that $Q^i = P(x)$. A1 is also not applicable, because if a non-empty tail of $T(x)$ matches that of Q, then it also matches that of $P(x)$. However, that is not possible after Step 1 has been applied.

■

The complexity of the *canonize* function is $O(m+n)$ because Step 1 uses a $O(n)$ algorithm that spans the transitory sequence and Step 2 can be performed by running the following $O(m)$ algorithm:

```
2.1. curr := 1; end := 1;
2.2. for i from 2 to |P(x)|
        if (P(x)[i] = P(x)[curr]) then
            if (curr = end) then curr := 1;
            else increment curr;
                if (i = |P(x)|) then end := |P(x)|;
                                     go to Step 2.3.;
        else end := i; curr := 1;
2.3. restrict P(x) to elements from 1 to end;
```

Canonization gives us an easy method to check if two signals are equivalent; canonize both signals, and perform a straight comparison of their transitory sequences and periods. The complexity of *canonize* is identical to the complexity of the minimization operation in Gaubert's work.

3.3 Homogeneous representation of signals

Consider the signals $x = ((4,3);(1,2))$ and $y = ((2);(3,1,5))$. It is difficult to compare these signals because their transitory sequences and periods have different lengths. It is easier to compare them by modifying their representations as follows: $x = ((4,3);(1,2,1,2,1,2))$ and $y = ((2,3);(1,5,3,1,5,3))$; hence, the following definition.

Definition 7 *The signals x and y are homogeneous to each other if $|T(x)| = |T(y)|$ and $PT(x)| = |P(y)|$.*

The following function converts two signals to their homogeneous forms:

```
homogenize(signal x,y)
{
    1. Canonize(x) and canonize(y);
    2. Assume that T(x) > T(y);
        Find U and P such that P(y) = U.P and |T(y).U| = |T(x)|;
            T(y) := T(y).U and P(y) := P.U;
    3. k := lcm(|P(x)|,|P(y)|); i := k/|P(x)|; j := k/|P(y)|;
        P(x) := P^i(x) and P(y) := P^j(y);
}
```

The following theorem proves the correctness of the *homogenize* function.

Theorem 3 - Homogeneity -

1. *The homogenize function returns homogeneous signals that are equivalent to its inputs.*
2. *There does not exist a pair of homogeneous signals equivalent to, and with shorter transitory sequences and shorter periods than the pair of homogeneous signals returned by the function homogenize.*

Proof: From Lemma 1, each step in the *homogenize* function preserves the equivalence of signals. It remains to be shown that neither A1 nor A2, in the definition of canonicity, can be applied to the signals obtained after the third step while still preserving their homogeneity. Clearly, A1 is not applicable since T_1 is the transitory sequence of the canonical form of x_1 and it remains unchanged by the *homogenize* function. A2 is also not applicable since we choose the smallest i and j such that $P^i(x) = P^j(y)$.

■

Let $n = |T(x) \oplus T(y)|$ and $m = |P(x) \oplus P(y)|$. Clearly, Step 1 is of $O(m+n)$ complexity, and Step 2 is of $O(n)$ complexity. Step 3 consists of the following three operations:

1. the computation of the least common multiple of $|P(x)|$ and $|P(y)|$ ($O(m^2)$ complexity),
2. the extension of $P(x)$ by a factor of at most $|P(y)|$ ($O(m)$ complexity), and
3. the extension of $P(y)$ by a factor of at most $|P(x)|$ ($O(m)$ complexity).

Therefore, the complexity of Step 3 is bounded by $O(m^2)$, and hence, the final complexity of *homogenize* is $O(m^2 + n)$.

3.4 Related Work

Though our definition is new, there exists a similar notation. In [9], Gaubert introduces a (max,+) algebra of formal series that are related to our definition of signals. A formal series is a polynomial

$$s = \bigoplus_{n,t \in Z} s(n,t)\gamma^n\delta^t$$

where Z is the set of integers, $s(n,t) = e$ or ε, and γ (δ) spans an index (time) space. The monom $\gamma^i\delta^j$ of a formal series s is equivalent, in our notation, to stating that the sequence S is such that $S[i] = j$. Moreover, formal series are a generalization of rational signals, which are polynomials in γ. Thus, the rational signal equivalent to the formal series $s = \bigoplus_{n,t>0} s(n,t)\gamma^n\delta^t$ is $s = \bigoplus_{n>0} t_n\gamma^n$. Furthermore, a series s is said to be periodic, if and only if there exist two polynomials p and q and a monomial r such that $s = p \oplus qr^*$. This representation is analogous to ours. However, Gaubert's algebra differs from ours in the definition of the delay operator as explained in Section 4.4.

4 Operations on Periodic Signals

In this section, we describe operations on signals for the algebraic representation of TEGs. We define a max operation to model synchronization, a backshift operator to represent the presence of initial tokens, and a delay operation to model delays.

4.1 Max operation

Definition 8 *Given two signals x and y, and their underlying sequences X and Y, the signal $z = \max(x, y)$, also denoted $z = x \oplus y$, is defined by its underlying sequence*

$$Z = \max(X, Y),$$

which is defined as the pointwise maximization of the sequences X and Y.

We now show that $\mathcal{Z}$ is closed under the max operation. Without loss of generality, we prove this result for homogeneous signals. Non-homogeneous signals can always be made homogeneous by using the *homogenize* function. Moreover, the closure of the max operation is quite obvious for signals with equal slope. This is however not the case for signals with different slopes. Intuitively, the next lemma states that, if two periodic signals x and y are such that the slope of x is steeper than the slope of y, then the signal $x \oplus y$ is equal to x for indices sufficiently large. The signal x is said to eventually dominate signal y. The lemma is best understood once the concept of distance of a periodic step to the slope of a signal is defined.

Definition 9 - distance of a periodic step to a slope -
For any periodic signal x, the distance $d(i)$ of a periodic step p_i (in $P(x)$) to the slope of x ($\sigma(x)$) is defined as follows:

$$\forall 1 \leq i \leq m : d(i) = \sum_{k=1}^{i} p_k - i\sigma(x)$$

Note that the distance of a periodic step to the slope of a signal x can be positive or negative. A positive (negative) distance indicates that the point corresponding to the periodic step in the spatial representation lies above (below respectively) a straight line defined by the equation $x_0 + i\sigma(x)$ where x_0 is the last point of the transitory sequence of x. We use the concept of distance in the next lemma to compute an upper bound of the index at which a signal effectively dominates another signal. For any periodic signal z, let $S(z)$ be the set of distances of the periodic steps to the slope of the signal z, i.e.,

$$S(z) = \{\sum_{k=1}^{i} p_k - i\sigma(z), 1 \leq i \leq m, p_i \in P(z)\}.$$

Lemma 2 *Let x and y be two homogeneous, periodic signals such that $T(x) = (x_0)$, $T(y) = (y_0)$, $|P(x)| = |P(y)| = m$, and $\sigma(x) > \sigma(y)$. Let X and Y be the underlying sequences of x and y respectively. Then,*

1. $\exists N : \forall k \geq N : (X \oplus Y)[k] = X[k]$,

2. *$x \oplus y$ is periodic, and*

3. *$N \leq \lceil \frac{\min S(x) - x_0 + \max S(y) + y_0}{\sigma(x) - \sigma(y)} \rceil$ where $S(x)$ is the set of distances of the periodic steps of x to its slope.*

Proof:

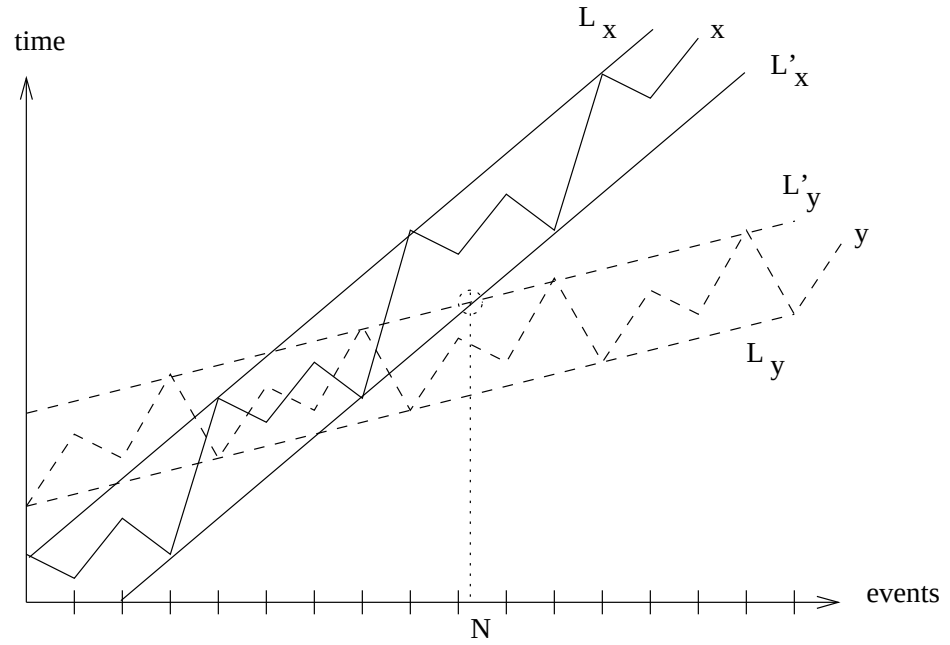

Figure 3: Maximization of two signals.

1. In the space representation, consider the straight line L_x (L_y respectively) passing through the points $X[k]$ ($Y[k]$ respectively) where $k \equiv 0 \pmod{m}$. Since $\sigma(x) > \sigma(y)$, L_x eventually lays above L_y in the space representation.

   Furthermore, since the elements of $P(x)$ ($P(y)$ respectively) are finite, all the points of x (y respectively) lay above (below respectively) a straight line L'_x (L'_y respectively) that is parallel to, but below L_x (above L_y respectively). Let N be the abscissa of the intersection point of L'_x and L'_y. Then, for any $k \geq \lceil N \rceil$, $X[k] > Y[k]$, and therefore, $(X \oplus Y)[k] = X[k]$.

2. According to Step 1 of this proof, $X \oplus Y$ is (but for a finite number of indices) equal to X. Since X is a periodic sequence, this shows that $X \oplus Y$ is a periodic sequence. Therefore, $x \oplus y$ is a periodic signal.

3. Observe that the equation of the straight line L'_x (L'_y respectively) defined in Step 1 of this proof is $X[k] = x_0 - \min S(x) + k\sigma(x)$ ($Y[k] = y_0 + \max S(y) + k\sigma(x)$ respectively). Therefore, the abscissa of the intersection point of L'_x and L'_y is given by the following equation:

$$y_0 + \max S(y) - x_0 + \min S(x) = k(\sigma(x) - \sigma(y));$$

   hence, the result in Statement 3 of the Lemma.

Figure 3 illustrates this proof by displaying the space representation of a signal x (and the associated lines L_x and L'_x) with a steeper slope than a signal y (and the associated lines L_y and L'_y). It also shows that N is bounded by the intersection of L'_x and L'_y.

■

Theorem 4 *$\mathcal{Z}$ is closed under finite application of max.*

Proof: We only need to prove the theorem for the maximization of two signals. Without loss of generality, assume that x and y are two homogeneous signals. If they are not homogeneous, take their homogeneous representations. Furthermore, assume that their transitory sequences have each n elements, and their periods have each m elements. We consider two cases.

Case 1 : $\sigma(x) = \sigma(y)$

Since the slope of x and y are equal, the following property holds:

$$\forall k > n, X[k] - Y[k] = X[k+m] - Y[k+m].$$

Therefore,

$$\forall k > n, (X[k] \oplus Y[k] = X[k]) \Rightarrow (X[k+m] \oplus Y[k+m] = X[k+m]),$$

and similarly,

$$\forall k > n, (X[k] \oplus Y[k] = Y[k]) \Rightarrow (X[k+m] \oplus Y[k+m] = Y[k+m]).$$

Therefore, $X \oplus Y$ is periodic, and so is $x \oplus y$.

Case 2 : $\sigma(x) \neq \sigma(y)$

Without loss of generality, assume that $\sigma(x) > \sigma(y)$. By applying Lemma 2 to the signals $x' = ((X[n]); P(x))$ and $y' = ((Y[n]); P(y))$, we can prove that $x' \oplus y'$ is a periodic signal, i.e., it consists of a finite transitory sequence and a period. Observe also that, for any $k > n$, $X[k] = X'[k-n]$ and $Y[k] = Y'[k-n]$. Therefore, we just prove that the restriction of $X \oplus Y$ to the event indices greater than n is a periodic sequence.

Since the restriction of $X \oplus Y$ to event indices less or equal to n is a finite sequence, this proves that $X \oplus Y$ is a periodic sequence. Therefore, $x \oplus y$ is a periodic signal.

■

The following algorithm computes the maximization of two periodic signals. Theorem 5 establishes its correctness.

maximize(x,y):
{
1. *Homogenize(x,y);*
2. *If* $\sigma(x) = \sigma(y)$ *then* $T(x \oplus y) = T(x) \oplus T(y)$,
   and $P(x \oplus y) = P(x) \oplus P(y)$;

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

}

Theorem 5 *The maximize function returns the maximization of its two input signals.*

Proof: The correctness of Step 2 of the *maximize* function follows from Case 1 of the proof of Theorem 4. The correctness of Step 3 and 4 of the *maximize* function follows from Case 2 of the proof of Theorem 4 and from Statement 3 of Lemma 2.

■

Let $n = |T(x)| \oplus |T(y)|$ and $m = |P(x)| \oplus |P(y)|$. The complexity of *maximize*(x,y) is $O(m^2 + n + N)$, where N is defined as in Statement 3 of Lemma 2.

Note that the infinite application of maximization does not necessarily result in a periodic signal. For example, consider the signal $z = ((); (1))$, which spans the set of natural numbers. Its underlying sequence is $Z = \{1, 2, 3, 4, 5, 6, 7, 8, 9, \ldots\}$. Now, consider the set $\mathcal{X}$ of signals x_k where the values for the first k indices of x_k's underlying sequence X_k are the sums of the k first natural numbers, and the values for the remaining indices are equal to $X_k[k]$, e.g., $X_1 = \{1, 1, \ldots\}$, $X_2 = \{1, 3, 3, \ldots\}$, and $X_3 = \{1, 3, 6, 6, \ldots\}$. Formally, $\mathcal{X}$ is defined as follows:

$$\mathcal{X} = \{x_k, k \ge 1 : x_k = ((Z[1], Z[1] + Z[2], \ldots, \sum_{i=1}^{k} Z[i]); (0))\}.$$

$\mathcal{X}$ is an infinite set of periodic signals; but the maximization of its elements $\bigoplus_{x \in \mathcal{X}} x$ corresponds to the sequence $\{1, 3, 6, 10, 15, 21, \ldots\}$, which cannot be represented with a periodic signal.

Our definition is the same as Gaubert's definition. Therefore our algorithm has the same complexity as Gaubert's algorithm. The complexity of both algorithms is dominated by the computation of the lowest common multiplier of the sizes of the periods.

4.2 Backshift operator

In a TEG, if a token is initially present in a given place, it immediately contributes to the enabling of the output transition of the place, even if the place's delay is not null. If several tokens are initially present in the same place, they fire one after the other without delays. Their firing order is inconsequential since tokens are anonymous.

The behavior of tokens present in the initial marking of a TEG is modeled by a backshift operator, represented by γ. Consider a transition corresponding to a specific event type. Assume that this transition has one input place that initially contains k tokens (this can be easily generalized to the case of a transition with multiple input places). Then, the first event affected by the delay of the input place has $k+1$ as an index. The first k events depend only on the initial time value associated with this particular event type, hence the following definition for γ.

Definition 10 *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 follows:*

$$\gamma x = \gamma(x) = ((\varepsilon, t_1, t_2, \ldots, t_n), P)$$

The presence of k tokens in an input place is modeled by the composition of k backshift operators, also noted γ^k. The main characteristics of the backshift operator are captured in the following theorem:

Theorem 6 - Characteristics of γ -

1. *$\mathcal{Z}$ is closed under the backshift operator $\gamma^i, \forall i \geq 1$;*
2. *the backshift operator distributes over the max operator, i.e.,*

$$\forall x, y \in \mathcal{Z}, \gamma(x \oplus y) = \gamma(x) \oplus \gamma(y).$$

Proof:

1. By definition, if x is a periodic signal, $P(\gamma x) = P(x)$, and $T(\gamma x) = \{\varepsilon\}.T(x)$. Therefore, both $P(x)$ and $T(x)$ are finite lists, and γx is a periodic signal. Statement 1 of Theorem 6 is true for $i = 1$.

   By recursive application of this result, we prove that $\mathcal{Z}$ is closed under finite application of γ. Moreover, since the signal corresponding to the sequence $\{\varepsilon, \varepsilon, \ldots\}$ is in $\mathcal{Z}$, $\mathcal{Z}$ is also closed under an infinite application of γ.

2. Let X and Y be the sequences underlying signals x and y.

$$\begin{aligned} \gamma(X \oplus Y) &= \{\varepsilon, X[1] \oplus Y[1], X[2] \oplus Y[2], \ldots\} \\ &= \{\varepsilon, X[1], X[2], \ldots\} \oplus \{\varepsilon, Y[1], Y[2], \ldots\} \\ &= \gamma(X) \oplus \gamma(Y) \end{aligned}$$

■

Example: Manufacturing Process

We now have enough information to build an algebraic description of 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 example is described by

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

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

4.3 Periodic delay function

This section defines and characterizes a function that can delay a periodic signal according to another periodic signal. We define the delay operation as a pointwise delay over the sequences underlying the input signal and the delay signal. However, a strict application of this definition could result in the loss of timing information in certain cases. For example, let $X = \{\varepsilon, 1, 2, 3, 4, \ldots\}$ be the sequence underlying the input signal and $D = \{\varepsilon, \varepsilon, 2, 2, 2, \ldots\}$ be the sequence underlying the delay signal. A brute force pointwise delay of X by D would result in the sequence $\{\varepsilon, \varepsilon, 4, 5, 6, \ldots\}$. The time value $X[2]$ has been lost in the process. The following definition avoids this type of problems.

Definition 11 - Delay in signal terms -
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,

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

Theorem 7 - Pointwise Addition -
Delays on signals correspond to pointwise addition on sequences.

Proof:

- $j \geq i$, i.e., there are at least as many ε's prefixing the delay signal as the input signal:

  Consider a periodic delay signal d and a periodic signal x such that the first non-ε element of $T(x)$ is at a higher index, i (in the transitory pattern) than the first non-ε element of $T(d)$. On one hand, the elements of $T(\delta_d(x))$ with indices strictly less than i are equal to ε. On the other hand, $\delta_d(X)[i] = X[i] + D[i]$ where $D[i]$ is the sum of the non-ε elements of $P(d)$ that have an index less or equal to i. After index i, elements in both X and D are different from ε, and therefore, they require only a simple pointwise addition.

- $j < i$, i.e., there are more ε's prefixing the input signal than the delay signal:

  Assume this time that the first non-ε element of $T(d)$ is at a higher index, j (in the transitory pattern) than the first non-ε element of $T(x)$. The only problem is then generated by the possible addition of some integers in $T(x)$ and ε's from $T(d)$. The ε's from $T(d)$ should be ignored; therefore, the $(j-1)^{th}$ elements of $T(x)$ are left unchanged. After index j, elements

in both X and D are different from ε, and therefore, they require only a simple pointwise addition.

Since there is no ε elements (by definition of a periodic signal) in neither $P(x)$ nor $P(d)$, the elements of $P(\delta_d(x))$ are simply the pointwise addition of the corresponding elements in $P(x)$ and $P(d)$.

■

For example, to delay the odd-indexed events of the signal $x = ((1);(2))$ by five and its even-indexed events by seven, one can delay x by δ_d where $d = ((5);(2,-2))$. Indeed, observe that

$$D = \{5,7,5,7,5,7,5,7,\ldots\}.$$

Therefore, performing a pointwise delay on the sequences D and X results in adding five to the odd-indexed events of X and seven to its even-indexed events. Note that signals d and x are homogeneous. If it is not the case, apply this result to the homogenized signals. Observe also that applying a constant delay function to a signal is quite simple as shown in the following corollary.

Corollary 1 *Let δ_d be a delay function with a constant delay signal $d = ((c);(0))$ and $x = ((t_1,\ldots,t_n);(p_1,\ldots,p_m))$ be a periodic signal. Then*

$$\delta_d(x) = ((t_1,\ldots,t_i+c,\ldots,t_n);(p_1,\ldots,p_m))$$

where t_i is the first non-ε element of $T(x)$.

Proof: Since $T(d)$ has no ε element, we apply Rule 2 of Definition 11.

■

Lemma 3 - Characteristics of the delay function -

1. *$\mathcal{Z}$ is closed under the operation defined by a periodic delay;*
2. *the composition of two periodic delay functions is commutative, associative, and the neutral element is the zero delay function;*
3. *the composition of a periodic delay function and the backshift operator is not commutative;*
4. *the composition of a constant delay function and the backshift operator is commutative.*

Proof:

1. Definition 11 proves that the obtained signal is periodic, and the closure of $\mathcal{Z}$ under the addition completes the proof.
2. We need to prove that the composition of delay functions is commutative, associative, and that its neutral element is the zero delay function.

Commutativity : let d and d' be two periodic delays defining two delay functions δ_d and $\delta_{d'}$. Let x be any periodic signal. We prove this property on the sequences D, D', and X. For any index k, without loss of generality, we only have to prove two cases.

(a) $D[k] \neq \varepsilon$ and $D'[k] \neq \varepsilon$

$$\begin{aligned}
\delta_{d'}(\delta_d(X))[k] &= \delta_{d'}(X+D)[k] \\
&= X[k] + D[k] + D'[k] \\
&= X[k] + D'[k] + D[k] \quad \text{(by commutativity of the addition)} \\
&= \delta_{d'}(X)[k] + D[k] \\
&= \delta_d(\delta'_d(X))[k]
\end{aligned}$$

(b) $D[k] = \varepsilon$

$$\begin{aligned}
\delta_{d'}(\delta_d(X))[k] &= \delta_{d'}(X)[k] \\
&= \delta_d(\delta'_d(X))[k]
\end{aligned}$$

Associativity : similar proof using the commutativity and associativity of the addition.

Neutral Element : the zero delay function is $\overline{0} = ((); (0))$, which can be written $((0); (0))$. By applying Corollary 1, we find that, for any periodic signal x, $\delta_{\overline{0}}(x) = x$.

3. Let $x = ((1); (2))$ be a periodic signal, and δ_d be a periodic delay function where $d = ((3); (1, -1))$. Then, on one hand,

$$\begin{aligned}
\delta_d(\gamma(x)) &= \delta_d((\varepsilon, 1); (2))) \\
&= ((\varepsilon, 5); (1, 3))),
\end{aligned}$$

but, on the other hand,

$$\begin{aligned}
\gamma(\delta_d(x)) &= \gamma((4); (3, 1))) \\
&= ((\varepsilon, 4); (3, 1))).
\end{aligned}$$

Therefore, $\delta_d(\gamma(x)) \neq \gamma(\delta_d(x))$.

4. Let x be a periodic signal, and δ_d be a constant delay function.

$$\begin{aligned}
\delta_d(\gamma(x)) &= \delta_d(((\varepsilon, t_1, \ldots, t_n); (p_1, \ldots, p_m))) \\
&= ((\varepsilon, t_1 + d, \ldots, t_n); (p_1, \ldots, p_m)) \\
&= \gamma((t_1 + d, \ldots, t_n); (p_1, \ldots, p_m)) \\
&= \gamma(\delta_d(x))
\end{aligned}$$

■

In some cases, it may be desirable to re-order an expression consisting of delays and backshifts in such a way that backshifts are applied before the delays. Even though the composition of a delay and a backshift is not commutative, it is possible to perform such a re-ordering using the following lemma.

Lemma 4 *Let d be a delay signal and x be an input signal. Then, $\gamma\delta_d(x) = \delta_{(\gamma d)}\gamma(x)$.*

Proof: Without loss of generality, we prove this result on homogenized signals.
Let $x = ((t_1, t_2, \ldots, t_n); (p_1, p_2, \ldots, p_m))$ and $d = ((u_1, u_2, \ldots, u_n); (q_1, q_2, \ldots, q_m))$.

$$\begin{aligned}
\delta_{(\gamma d)}\gamma(x) &= \delta_{((\varepsilon,u_1,u_2,\ldots,u_n);(q_1,q_2,\ldots,q_m))}[((\varepsilon, t_1, t_2, \ldots, t_n); (p_1, p_2, \ldots, p_m))] \\
&= ((\varepsilon, t_1 + u_1, t_2 + u_2, \ldots, t_n + u_n); (p_1 + q_1, p_2 + q_2, \ldots, p_m + q_m)) \\
&= \gamma[((t_1 + u_1, t_2 + u_2, \ldots, t_n + u_n); (p_1 + q_1, p_2 + q_2, \ldots, p_m + q_m))] \\
&= \gamma[\delta_d(x)]
\end{aligned}$$

■

Using Definition 11, we define the following algorithm to compute the periodic delay of a signal:

```
delay (signal x, signal d): signal y
{
    1. Homogenize(x,d);
    2. init := 0;
    3. for i from 1 to n do
        if (T(x)[i] = ε) then
            T(y)[i] := ε;
            if (T(d)[i] ≠ ε) then
                init := init + T(d)[i];
        elsif (T(d)[i] = ε) then
            T(y)[i] := T(x)[i];
        else
            T(y)[i] := T(x)[i] + T(d)[i] + init;
            init := 0;
    4. P(y) := pointwise addition of P(x) and P(d);
    5. Canonize (y);
}
```

Recall that the complexity of the *homogenize* function is $O(m^2 + n)$ where $m = |P(x)| \oplus |P(d)|$ and $n = |T(x)| \oplus |T(d)|$. Steps 2 and 3 is of complexity $O(m + n)$ while Step 4 is of complexity $O(m)$. Since the complexity of the *canonize* function is $O(m + n)$, the complexity of the *delay* function is $O(m^2 + n)$.

Example: Manufacturing Process

In our manufacturing process example, the 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 $(();(4,-3,0,0,0))$.

4.4 Related Work

As mentioned in the Section 1, our work is closely related to Gaubert's work on (max,+) algebra [9]. This section analyzes the differences in the basic operators of both algebras.

There is no fundamental differences between Gaubert's maximization operator and ours. Therefore, it is not surprising that our implementation of maximization is similar to Gaubert's implementation. Both implementations are based on the ability to find an upper bound to the last index for which two signals of different slopes may intersect. In both algebras, the computation of this upper bound can be done in $O(m+n)$ where n (m) is the length of the transitory sequence (period respectively) of the homogenized signals. Similarly, in both algebras, the computation of a backshifted signal is immediate since it is merely a matter of increasing all indices by the amount of backshift applied to the signal.

Our definition of the delay function ($\otimes$) differs from Gaubert's definition [9]. We define the delay of a signal by another signal as the pointwise delay of the sequences underlying the two signals while Gaubert defines it as the sup-convolution of the two underlying sequences. A delay is nothing more than the "multiplication" of two signals. Since Gaubert represents signals as polynomials, it is natural to define their "multiplication" as a convolution. Let x and y be two signals and X and Y be their respective underlying sequences. Bearing some index translation, Gaubert's delay ($\otimes_G$) can be expressed as follows.

$$\forall i > 0 : (X \otimes_G Y)[i] = \bigoplus_{j=1}^{i} X[j]Y[i+1-j].$$

We now show that Gaubert's delay can be modeled using our delay, and that the converse is not true. We restrict our analysis to strictly monotonic periodic signals to match Gaubert's definition.

Lemma 5 *Let x and y be any two strictly monotonic periodic signals. There exists a TEG, governed under operations of the (max,+) algebra of signals, that models $x \otimes_G y$.*

Proof:
Assume that n (m) is the size of the transitory sequence (period respectively) of signal y and that $p = \sum_{k=1}^{m} p_k$ where the p_k's (for k in $1, \ldots, m$) are the periodic steps of y. As illustrated in Figure 4, let τ_x be the transition associated with signal x. Define two transitions τ_t and τ_p as follows. τ_t is the synchronization transition for n input places $T_1, \ldots, T_n$ that have τ_x as a common input transition and such that each T_i has $i-1$ initial tokens and a constant delay signal $((Y[i]);(0))$. Similarly, τ_p is the synchronization transition for m input places $P_1, \ldots, P_m$ that have τ_x as a common input transition and such that each P_j has $j+n-1$ initial tokens and a constant delay signal $((Y[j+n]);(0))$. However, unlike τ_t, τ_p also has a feedback control loop with a place P that has $n+m$ initial tokens and a constant delay signal $((p);(0))$. Note that x is an input transition common to all the T_i's and P_j's. Moreover, connect τ_t and τ_p (each via a place with no initial tokens and a null constant delay) to a common transition that generates a signal called z. In Figure 4, initial tokens for a place are indicated by a number at the center of the circle representing the place.

We now show that $z = x \otimes_G y$. As indicated in Figure 4, let u (v) be the signal generated by transition τ_t (τ_p). Let also X, Y, Z, U, and V be the underlying sequence of x, y,z,u, and v. Then,

$$U[i] = \begin{cases} \bigoplus_{k=1}^{i} X[i+1-k]Y[k] & \text{for } 1 \leq i \leq n \\ \\ \bigoplus_{k=1}^{n} X[i+1-k]Y[k] & \text{for } i \geq n+1 \end{cases}$$

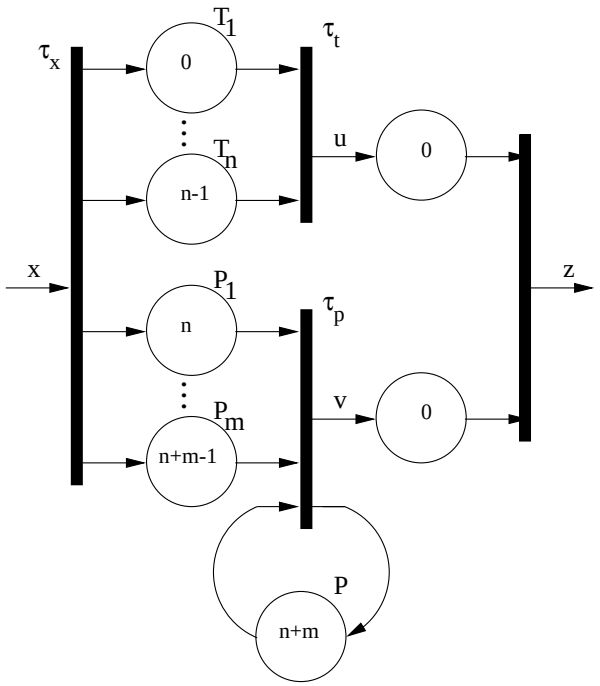

Figure 4: Timed Event Graph modeling the expression $z = x \otimes_G y$.

Moreover,

$$V[i] = \begin{cases} \varepsilon & \text{for } 1 \leq i \leq n \\ \bigoplus_{k=n+1}^{i} X[i+1-k]Y[k] & \text{for } n+1 \leq i \leq n+m \\ \bigoplus_{k=n+1}^{i} X[i+1-k]Y[k] & \text{for } n+m+1 \leq i \text{ (because } Y[n+m+k] = Y[n+k] + P \text{ for } j \geq 1) \end{cases}$$

The proof is completed by observing that $X \oplus_G Y = U \oplus V$, and therefore, the TEG described in Figure 4 models $x \oplus_G y$.

■

Lemma 5 shows that any expression in Gaubert's algebra can be modeled with the (max,+) algebra of signals. The next lemma shows that the converse is not true.

Lemma 6 - Expressiveness -

1. *For any two strictly monotonic periodic signals x and y (with no ε transitory steps), there exists a periodic signal d such that $\delta_d(x) = y$.*

2. *There exist three strictly monotonic periodic signals x, y and z (with no ε transitory steps) such that $x \otimes_G z = y$.*

Proof:

1. Suppose that x and y are homogeneous. If it is not the case, homogenize them. Then, define the periodic signal d such that each of its (transitory and periodic) steps is the arithmetic difference between the corresponding steps in y and x. Using Definition 11, it is easy to verify that $\delta_d(x) = y$.

2. Let $X = \{1, 3, 5, 7, 9, \ldots\}$ and $Y = \{3, 4, 7, 8, 11, 12, \ldots\}$. Suppose there exists a signal z such that $x \otimes_G z = y$. This implies that $Z[1] = 2$. Therefore, $Y[2] \geq X[2] + Z[1] = 5$, which is a contradiction since $Y[2] = 4$. Therefore, there cannot exist a strictly monotonic periodic signal z such that $x \otimes_G z = y$.

■

5 Closure Operations

Recall that A^*v is the least solution of $x = Ax \oplus v$ [1]. This section describes algorithms that help compute A^* where $A^* = \bigoplus_{k \geq 0} A^k$.

5.1 Periodic *-delay function

Computing A^*v requires the computation of expressions of the form $\bigoplus_{k \geq 0} (\delta_d \gamma)^k$ where $\delta_d \gamma$ is the composition of the periodic delay function δ_d and the backshift function. Note also that $(\delta_d \gamma)^k$ stands for the composition of k factors equal to $\delta_d \gamma$ (i.e., $\delta_d \gamma = \gamma \circ \delta_d$). We refer to such an expression as a periodic *-delay function. We also use the abbreviated form $(d\gamma)^*$, in which the delay function is represented only by its delay signal.

Definition 12 *A periodic *-delay function $(d\gamma)^*$ is defined by a periodic delay signal d as follows:*

$$(d\gamma)^* = \bigoplus_{k \geq 0} (\delta_d \gamma)^k$$

where the slope of d is null.

As mentioned in the previous section, maximizing an infinite number of periodic signals does not necessarily yield a periodic signal. We show that the *-delay function yields a periodic signal by proving that $\bigoplus_{k \geq 0} (\delta_d \gamma)^k$ is equal to the maximization of a finite set of signals. First, note that, if n is the length of the transitory period of the delay signal d,

$$\bigoplus_{k \geq 0} (\delta_d \gamma)^k = [\bigoplus_{k \geq 0}^{k=n-1} (\delta_d \gamma)^k] \oplus (\delta_d \gamma)^n [\bigoplus_{i \geq 0} (\delta_d \gamma)^i].$$

Therefore, it suffices to show that a *-delay function yields a periodic signal for a delay signal with only one transitory step (i.e., the underlying sequence of the delay consists of a periodic sequence). We start by showing that $(d\gamma)^*$ can be written as the infinite maximization of periodic signals that have the same period. From now on, define $\{x_i, i > 0\}$ as a family of periodic signals such that

$$x_i \quad = \quad \gamma^{i-1}((X[i]); (D[i+1], \ldots, D[i+m])). \tag{1}$$

Lemma 7 *Let d be a periodic signal such that $|T(d)| = 1$ and $\sigma(d) = 0$. Let x be any periodic signal. Then,*

$$(d\gamma)^*(x) = \bigoplus_{i>0} x_i$$

where x_i is defined as in (1).

Proof:
Let G be the underlying sequence of $\bigoplus_{i>0} x_i$. Then,

$$\begin{aligned}
\forall k \geq 1 : G[k] &= \bigoplus_{i=1}^{k} X_i[k] && \text{(because, for all } i > k,\ X_i[k] = \varepsilon) \\
&= \bigoplus_{i=1}^{k} (X[i] + \sum_{j=i+1}^{k} D[j]) && \text{(by definition of the } x_i)
\end{aligned}$$

Observe that $(X[i] + \sum_{j=i+1}^{k} D[j])$ is the value at index k of the sequence underlying $(\delta_d \gamma)^l$ where $l = k - i$. Therefore, the underlying sequences of $(d\gamma)^*(x)$ and $\bigoplus_{i>0} x_i$ are equal; hence, the lemma.

■

It is easy to verify that the x_i's have the same period, and therefore, the same slope. We now establish two results on the x_i's based on the comparison of their slope and the slope of an input signal.

Lemma 8 - Slope Characterization -
Let X be the underlying sequence of a periodic signal x and $|T(x)| = n$ and $|P(x)| = m$.
Let D be the underlying sequence of a periodic signal d such that $|T(d)| = 1$, $|P(d)| = m$, and $\sigma(d) = 0$
Let $x_i, i > 0$ be defined as in (1). For any $i > n$ and any $k > n$,

1. $(\sigma(x_i) \geq \sigma(x)) \Rightarrow X_{i+m}[k] \oplus X_i[k] = X_i[k]$
2. $(\sigma(x_i) < \sigma(x)) \Rightarrow X_{i+m}[k] \oplus X_i[k] = X_{i+m}[k]$

Proof: Remember that, for any i and k, $X_i[k] = X[i] + \sum_{j=i+1}^{k} D[j]$. Then, for $i > n$,

$$\begin{aligned}
X_{i+m}[k] - X_i[k] &= X[i+m] + \sum_{j=i+m+1}^{k} D[j] - (X[i] + \sum_{j=i+1}^{k} D[j]) \\
&= X[i+m] - X[i] - \sum_{j=i+1}^{j=i+m} D[j] \\
&= m[\sigma(x) - \sigma(x_i)] && \text{(because } |T(x)| = |T(d)| = m)
\end{aligned}$$

The results in Lemma 8 follow directly from this equality.

■

We now have enough elements to prove

Theorem 8 *The signal resulting from the application of a periodic *-delay function to a periodic input signal of null slope is a periodic signal.*

Proof: As previously mentioned, the problem can be reduced to proving that $\bigoplus_{k\geq 0}(\delta_d\gamma)^k$ where d has a single transitory step and a null slope. Let x be an input periodic signal such that $|T(x)| = n$ and $|P(x)| = m$). Assume also that $|P(d)| = m$. Define $\{x_i, i > 0\}$ as previously. Let D and X be the underlying sequences of d and x respectively. From Lemma 7,

$$(d\gamma)^*(x) = \bigoplus_{i>0} x_i.$$

As in Lemma 8, we consider two cases.

1. $\sigma(x_i) \geq \sigma(x)$

   Using Lemma 8,

   $$\forall p < m : \bigoplus_{i>0} x_{n+im+p} = x_{n+p},$$

   hence,

   $$\bigoplus_{i>0} x_i = \bigoplus_{i=n+1}^{n+m} x_i.$$

   Therefore, according to Theorem 4, $(d\gamma)^*(x)$ is a periodic signal.

2. $\sigma(x_i) < \sigma(x)$

   Let Y be the underlying sequence of $(d\gamma)^*(x)$. Because $X_i[k] = \varepsilon$ for any $i > k$,

   $$\forall k > n + m : Y[k] = \bigoplus_{i=n+1}^{k} X_i[k].$$

   Using Lemma 8,

   $$\forall k > n + m : Y[k] = \bigoplus_{i=k-m}^{k} X_i[k].$$

   Since $X_{i+m}[k] - X_i[k]$ is constant (see proof of Lemma 8), Y is periodic and $(d\gamma)^*(x)$ is a periodic signal.

■

The *star-delay* algorithm (for computing the *-delay of a periodic signal) derives from the proof of Theorem 8. Its description is followed by its proof of correctness.

star-delay (signal x, delay d): signal y
{
    1. Homogenize (x,d);

2. *Compute* $\sigma(x_i) = 1/m \sum_{i=n+1}^{m} D[i]$;
3. *if* $(\sigma(x_i) \geq \sigma(x))$ *then*
   $y := \bigoplus_{k=1}^{n+m} x_k$;
4. *else*
   Build z such that
      $T(z) :=$ *elements from* $n+1$ *and* $n+2m$ *of* $\bigoplus_{k=n+1}^{n+2m} x_k$;
      $P(z) :=$ *elements from* $n+2m+1$ *and* $n+3m$ $\bigoplus_{k=n+m}^{n+3m} x_k$;
   $y :=$ *Maximize*$(z, \bigoplus_{k=1}^{n} x_k)$;
5. *Canonize (y);*

}

Lemma 9 *The function star-delay is correct.*

Proof: The correctness of the path consisting of Steps 1,2,3, and 5 of the algorithm is given by the case corresponding to $(\sigma(x_i) \geq \sigma(x))$ in the proof of Theorem 8.

The correctness of the path consisting of Steps 1,2,4, and 5 of the algorithm is obtained as follows.First, we prove the correctness of the formula for z. Case 2 $(\sigma(x_i) < \sigma(x))$ in the proof of Theorem 8 shows that the period of z starts at the latest at index $n + 2m + 1$, hence the formula for $T(z)$ and $P(z)$. Furthermore, since

$$(d\gamma)^*(x) = \bigoplus_{j=1}^{j=n} x_j \oplus (\bigoplus_{k>n} x_k)$$

and

$$z = \bigoplus_{k>n} x_k,$$

then,

$$(d\gamma)^*(x) = \bigoplus_{j=1}^{j=n} x_j \oplus z.$$

■

The complexity of Step 1 is $O(m^2+n)$. The complexity of Step 2 of the algorithm is $O(m)$. The complexity of Step 3 is $O((n+m)(m^2+n+N))$ where m^2+n+N is the cost of maximizing two signals. Similarly, the complexity of Step 4 of the algorithm is $O((n+m)(m^2+n+N))$. Finally, the complexity of Step 5 is $O(m+n)$. Therefore, the overall complexity of the **-delay* function is $O((n+m)(m^2+n+N))$.

5.2 Closure matrices of signals

The closure of matrices (of periodic signals) can be obtained by the application of Jordan's algorithm (which is a variant of Gauss's algorithm). Various implementations are described in Gondran and Minoux [11].

Jordan (matrix A): matrix A^*
{

1. $A^{(0)} = A$;
2. *For all* $k = 1, \ldots, n$:
   for all $i, j = 1, \ldots, n$:
   $A_{ij}^{(k)} = A_{ij}^{(k-1)} \oplus A_{ik}^{(k-1)} (A_{kk}^{(k-1)})^* A_{kj}^{(k-1)}$;
3. $A^* = A^{(n)} \oplus I$;

}

The correctness of this algorithm is well known, and its complexity is $O(n^3)$ where n is the number of entries in the transition matrix (which correspond to the number of events in the system).

Even though the transition matrix of a DES is in general sparse, its closure matrix is always quite dense. Therefore, computing the closure matrix might not always be practical. There are many algorithms that avoid computing the full closure matrix. Such algorithms are described in [11]. In a future work, we will explore means of reducing complexity, including a decomposition technique based on strongly connected components.

Example: Manufacturing Process

Using Jordan's algorithm, we compute A^*, the closure of the matrix A that characterizes our manufacturing process model.

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

It corresponds to the matrix computed by Cofer and Garg, and yields

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

which corresponds to the following infinite periodic sequence:

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

Manually computing these infinite sequences is a tedious, error-prone task. Our implementation negates this problem, and empowers the user with the ability of focusing on the real problems.

6 Controllability

This section describes how the (max,+) algebra of signals can test the controllability of a specification with respect to a DES expressed as a TEG. This work is based on the work of Cofer and Garg [3, 4] on the controllability of DES using (max,+) algebra. We first define the concept of a specification and its controllability; then, we describe how to compute supremal controllers for a set of of extremal behaviors. Since infimal controllers may not exist for some types of specifications, we refrain from discussing them.

Specifying acceptable behaviors in (max,+) algebra is done by defining finite (or infinite) sets of vectors where the components of each vector represent lower/upper bounds for the signals generated by the system. For example, a vector specifying an upper bound can be used to impose timing constraints on the event occurrences in a system. In general, not all events in a system are controllable. In the (max,+) algebra of signals, controllable events are described by a matrix I_c in which non-diagonal elements and uncontrollable elements on the diagonal are equal to ε and controllable elements on the diagonal are equal to the zero signal $(();(0))$.

Let A be the transition matrix describing a system in which the controllable events are described by the matrix I_c. Let v be the vector describing the initial conditions. Assume that the set of acceptable behaviors is Y. Then, Cofer shows that the controllability of Y with respect to an upper bound y in Y can be computed using the following equation:

$$A^*(I_c y \oplus v) \leq y.$$

Similarly, if y is a lower bound, then the controllability of Y is given by the following equation:

$$A^*(I_c y \oplus v) \geq y.$$

Assume that, in the manufacturing process example, the set of specifications Y is the set of signals less than or equal to

$$y = \begin{pmatrix} ((0);(7)) \\ ((1);(7)) \\ ((5);(7)) \end{pmatrix}$$

To determine if Y is controllable, we compute

$$A^*(I_c y \oplus v) = \begin{pmatrix} ((0);(5,7)) \\ ((1,7);(7,7,7,8,6)) \\ ((5,7);(7,7,7,8,6)) \end{pmatrix}$$

which shows that Y is not controllable since $((5,7);(7,7,7,8,6))$ is not less than or equal to $((5);(7))$.

Cofer and Garg also show that, even if a specification is not controllable, a controllable specification can always be found. Similarly, one might wish to compute the greatest superset of behaviors which is invariant under uncontrollable actions. Cofer and Garg describe algorithms to compute such supremal controllers for four types of specifications. If the specification is a finite set $Y = \{y_1, \ldots, y_k\}$ of behaviors, the supremal controller set is given by

$$\{x \in Y, A^*(I_c x \oplus v) \in Y\}$$

As shown in the next section, we implemented this type of extremal behavior specification, and we were able to show that if the set Y of specification is as follows:

$$Y = \left\{ \begin{pmatrix} ((0);(7)) \\ ((1);(7)) \\ ((5);(7)) \end{pmatrix}, \begin{pmatrix} ((0);(8)) \\ ((1);(8)) \\ ((5);(8)) \end{pmatrix} \right\}$$

the supremal set is given by

$$\left\{ \begin{pmatrix} ((0);(8)) \\ ((1);(8)) \\ ((5);(8)) \end{pmatrix} \right\}.$$

7 Implementation

We validated the algorithms described in previous sections by implementing them in C++. In this section, we present a brief introduction to the resulting tool.

The tool allows the definition of a DES in the form of its transition matrix. Elements of the matrix are defined as expressions. Simple expressions include ε, delay functions, and the backshift function. A complex expression consists of the combination of maximization, delay, or star-delay operations on simple expressions. For example, the following commands describe the matrix for our manufacturing process example. Note that the indices in the matrix start at zero (since arrays are indexed from zero in C++).

```
expression a = delay ((7);(-2,2));
expression w = delay ((4);(0));
expression s = delay ((1);(0));
expression r = delay ((4);(-3,0,0,0,3));
matrix A[3];
A[0,0] = a.Y;
A[1,0] = s;
A[2,1] = w;
A[1,2] = r.Y;
```

The closure matrix of a transition matrix is calculated using Jordan's algorithm. Its application to a vector describing the initial conditions of the DES computes the least solution of the equation $x = Ax \oplus v$. Thus, the following commands computes the solution A^*v. Note that the initial vector is initialized with the expression `e`, which denotes the null signal $(();(0))$.

```
matrix Astar = closure (A);
vector V = e[3];
solve X = Astar.V;
```

The tool can also compute the controllability of a DES. Thus, the controllability of our manufacturing process example (with respect to the upper bound specification y given that only Event t_2 is controllable) can be tested by the following command. Note that the index of Event t_2 in the transition matrix is one.

```
controllable (1) upper y wrt Astar V;
```

Supremal controllers can also be computed for four types of specifications: finite set of behaviors, range of behaviors, single sequence behavior, and behaviors based on matrices of defining separation times. The following commands illustrate the computation of a supremal controller for a set of two behaviors. Note that all three events are defined as being controllable.

```
specification y = set y0, y1;
```

```
display specification y;
supremal S = y (0,1,2) wrt Astar V;
```

8 Conclusion

We have defined a (max,+) algebra of periodic signals that mechanizes the computation of supremal controllers for timed discrete event systems. Signals are finite representations of infinite event sequences that exhibit a periodic pattern. We defined several algorithms to implement basic operations such as synchronization, delay and closure operations. We also proved the correctness of each algorithm and characterized their complexities. We showed that, although our algebra is similar to Gaubert's (max,+) algebra [9], it is more expressive because we use a pointwise delay instead of a delay defined as the sup-convolution of two signals. Future work will explore means of reducing complexity in the computation of the closure matrix and applications to traditional real-time problems.

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] D. D. Cofer. *Control and Analysis of Real-Time Discrete Event Systems.* PhD thesis, The University of Texas at Austin, 1995.

[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. *IEEE Transactions on Automatic Control*, 41(2):199–209, February 1996.

[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] S. Gaubert. Methods and applications of (max,+) linear algebra. Technical Report 3088, Institut National de Recherche en Informatique et en Automatique, January 1997.

[11] M. Gondran and M. Minoux. *Graphs and Algorithms*. John Wiley and Sons, 1986.

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