Chapter 21
Discrete-Event Systems in a Dioid Framework: Modeling and Analysis

Thomas Brunsch, Jörg Raisch, Laurent Hardouin, and Olivier Boutin

21.1 Timed Event Graphs

As discussed in the previous chapters, a Petri net graph is a directed bipartite graph $N = (P,T,A,w)$, where $P = \{p_1,\ldots,p_n\}$ is the finite set of places, $T = \{t_1,\ldots,t_m\}$ is the (finite) set of transitions, $A \subseteq (P \times T) \cup (T \times P)$ is the set of directed arcs from places to transitions and from transitions to places, and $w : A \to \mathbb{N}$ is a weight function. Note that this notation differs slightly from the notation introduced in Part II of this book. However, for the description of timed event graphs, i.e., a specific type of Petri nets, our notation is very convenient. In the sequel, the following notation is used for Petri net graphs:

$$^\bullet t_j := \{p_i \in P \,|\, (p_i,t_j) \in A\}$$

Thomas Brunsch · Jörg Raisch
Technische Universität Berlin, Fachgebiet Regelungssysteme, Einsteinufer 17,
10587 Berlin, Germany
e-mail: {brunsch,raisch}@control.tu-berlin.de

Jörg Raisch
Fachgruppe System- und Regelungstheorie,
Max-Planck-Institut für Dynamik komplexer technischer Systeme,
Sandtorstr. 1, 39106 Magdeburg, Germany

Laurent Hardouin · Thomas Brunsch
LUNAM, University of Angers, LISA, ISTIA, 62 Av. Notre-Dame du Lac,
49000 Angers, France
e-mail: laurent.hardouin@istia.univ-angers.fr

Olivier Boutin
Calle Santiago 2 – 4ºC, 11005 Cadiz, Spain
e-mail: olivier.research@gmail.com

C. Seatzu et al. (Eds.): Control of Discrete-Event Systems, LNCIS 433, pp. 431–450.
springerlink.com © Springer-Verlag London 2013

is the set of all input places for transition t_j, i.e., the set of places with arcs to t_j.

$$t_j{}^\bullet := \{p_i \in P \,|\, (t_j, p_i) \in A\}$$

denotes the set of all output places for transition t_j, i.e., the set of places with arcs from t_j. Similarly,

$${}^\bullet p_i := \{t_j \in T \,|\, (t_j, p_i) \in A\}$$

is the set of all input transitions for place p_i, i.e., the set of transitions with arcs to p_i, and

$$p_i{}^\bullet := \{t_j \in T \,|\, (p_i, t_j) \in A\}$$

denotes the set of all output transitions for place p_i, i.e., the set of transitions with arcs from p_i. Obviously, $p_i \in {}^\bullet t_j$ if and only if $t_j \in p_i{}^\bullet$, and $t_j \in {}^\bullet p_i$ if and only if $p_i \in t_j{}^\bullet$. Graphically, places are shown as circles, transitions as bars, and arcs as arrows. The number attached to an arrow is the weight of the corresponding arc. Usually, weights are only shown explicitly if they are different from one.

A Petri net system (or Petri net) is a pair $(N, \boldsymbol{m}^0)$, where $N = (P, T, A, w)$ is a Petri net graph and $\boldsymbol{m}^0 \in \mathbb{N}_0^n$ with $n = |P|$ is a vector of initial markings. In graphical representations, the vector of initial markings is shown by m_i^0 dots ("tokens") within the circles representing places p_i, $i = 1, \ldots, n$. A Petri net can be interpreted as a dynamical system with a state signal $\boldsymbol{m} : \mathbb{N}_0 \to \mathbb{N}_0^n$ and an initial state $\boldsymbol{m}(0) = \boldsymbol{m}^0$. Its dynamics is governed by two rules, also called firing rules:

(i) In state $\boldsymbol{m}(k)$, a transition t_j can occur (or "fire") if and only if all of its input places contain at least as many tokens as the weight of the arc from the respective place to the transition t_j, i.e., if

$$m_i(k) \geq w(p_i, t_j), \forall p_i \in {}^\bullet t_j.$$

(ii) If a transition t_j fires, the number of tokens in all its input places is decreased by the weight of the arc connecting the respective place to the transition t_j , and the number of tokens in all its output places is increased by the weight of the arc connecting t_j to the respective place, i.e., the state changes according to

$$m_i(k+1) = m_i(k) - w(p_i, t_j) + w(t_j, p_i),\ i = 1, \ldots, n,$$

where $m_i(k)$ and $m_i(k+1)$ represent the numbers of tokens in place p_i before and after the firing of transition t_j.

Note that a place can simultaneously be an element of ${}^\bullet t_j$ and $t_j{}^\bullet$. Hence the number of tokens in a certain place can appear in the firing condition for a transition whilst being unaffected by the actual firing. It should also be noted that a transition enabled to fire might not actually do so. In fact, it is well possible that, in a certain state, several transitions are enabled simultaneously, and that the firing of one of them will disable the other ones.

A Petri net $(N, \boldsymbol{m}^0)$ is called an event graph (or synchronization graph), if each place has exactly one input transition and one output transition, i.e.,

$$|{}^\bullet p_i| = |p_i{}^\bullet| = 1, \forall p_i \in P,$$

and if all arcs have weight 1. It is obvious that an event graph cannot model conflicts or decisions, but it does model synchronization effects. A small example of an event graph is given in Fig. 21.1.

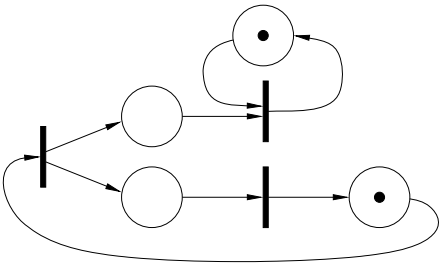

Fig. 21.1 Example of an event graph

A standard Petri net $(N, \boldsymbol{m}^0)$ only models the ordering of firings of transitions, but not the actual firing times. However, it is possible to "attach" timing information to the "logical" DES model $(N, \boldsymbol{m}^0)$. This can be done in two ways: time can be associated with transitions (representing transition delays) or with places (representing holding times). In timed event graphs (TEG), transition delays can always be transposed into holding times by simply shifting each transition delay to all input places of the corresponding transition. However, it is in general not possible to convert holding times into transition delays. Therefore, we will only consider timed event graphs with holding times. In TEG with holding times, tokens in place p_i have to be held for a certain time (called "holding time") before they can contribute to the firing of the output transition of p_i. The holding time for a token in place p_i is denoted v_i.

Figure 21.2 shows a part of a general timed event graph with holding times. In general, the earliest time instant when place p_i receives its k^{th} token is denoted $\pi_i(k)$,

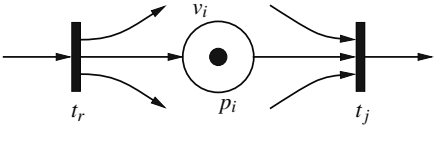

Fig. 21.2 Part of a general timed event graph with holding times

and $\tau_j(k)$ denotes the earliest time instant when transition t_j can fire for the k^{th} time. Then the earliest time of the k^{th} firing of transition t_j can be determined by

$$\tau_j(k) = \max_{p_i \in {}^\bullet t_j} (\pi_i(k) + v_i),$$

i.e., a transition is enabled to fire as soon as all input places of this transition have received their k^{th} token and their corresponding holding times have elapsed. Similarly, the earliest time instant when place p_i receives its $(k + m_i^0)^{\text{th}}$ token can be determined by the k^{th} firing of its input transition, i.e., $\pi_i(k + m_i^0) = \tau_r(k), t_r \in {}^\bullet p_i$. Note that a place in an event graph has exactly one input transition and consequently, the place can only receive tokens when this input transition fires. Therefore, it is possible to eliminate the π_i to give recursive equations for the (earliest) firing times of transitions.

21.2 Motivational Example

From the previous section, it is clear that it is possible to recursively compute the earliest possible firing times for transitions in timed event graphs. In the corresponding equations, two operations were needed: maximization and addition. To illustrate this, a small example (taken from Cassandras, Lafortune & Olsder [4]) is used. Imagine a simple public transportation system consisting of three lines: an inner loop and two outer loops. There are two stations where passengers can change lines, and four rail tracks connecting the stations. The basic structure of the system is given in Fig. 21.3. Initially, it is assumed that the train company operates one train on each track. A train needs 3 units of time to travel from station 1 to station 2, 5 units of time for the track from station 2 to station 1, and 2 and 3 units of time for the outer loops, respectively. The aim is to implement a user-friendly policy where trains wait for each other at the stations to allow passengers to change lines without delay, i.e., the departure times of trains from stations shall be synchronized. This can be easily represented in a timed event graph with holding times (see Fig. 21.4). The tokens

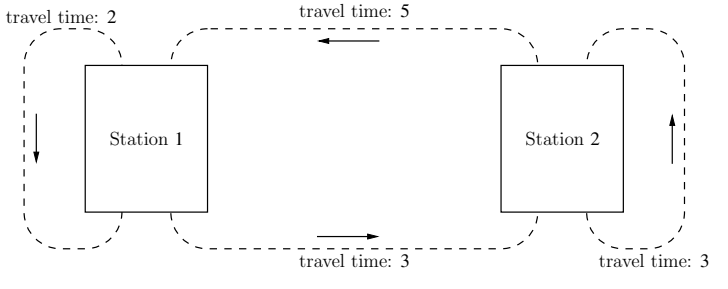

Fig. 21.3 Simple transportation network taken from [4]

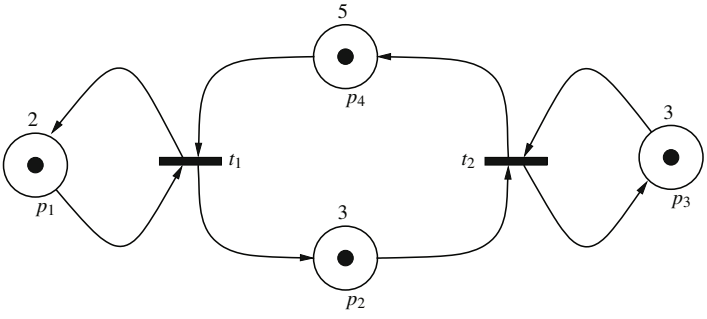

Fig. 21.4 Timed event graph of the transportation network

in places p_1 to p_4 model trains on the tracks and the holding times are the traveling times for trains on the tracks. The firing times of transitions t_1 and t_2 represent the departure times of the trains in stations 1 and 2, respectively. These times can therefore be interpreted as the "time table" for the simple public transportation system. The recursive equations for the firing times of transitions t_1 and t_2 are

$$\begin{aligned} \tau_1(k) &= \max\left(\pi_1(k)+2, \pi_4(k)+5\right) \\ \tau_2(k) &= \max\left(\pi_2(k)+3, \pi_3(k)+3\right) \end{aligned} \tag{21.1}$$

and

$$\pi_1(k+m_1^0) = \pi_1(k+1) = \tau_1(k) \tag{21.2}$$
$$\pi_2(k+m_2^0) = \pi_2(k+1) = \tau_1(k) \tag{21.3}$$
$$\pi_3(k+m_3^0) = \pi_3(k+1) = \tau_2(k) \tag{21.4}$$
$$\pi_4(k+m_4^0) = \pi_4(k+1) = \tau_2(k). \tag{21.5}$$

Inserting (21.2)–(21.5) into (21.1), it is possible to eliminate $\pi_i(k)$

$$\begin{aligned} \tau_1(k+1) &= \max\left(\tau_1(k)+2, \tau_2(k)+5\right) \\ \tau_2(k+1) &= \max\left(\tau_1(k)+3, \tau_2(k)+3\right). \end{aligned} \tag{21.6}$$

Now, given initial firing times, i.e., the first departures of trains, $\tau_1(1) = \tau_2(1) = 0$, the timetable can be determined as follows:

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 5 \\ 3 \end{bmatrix}, \begin{bmatrix} 8 \\ 8 \end{bmatrix}, \begin{bmatrix} 13 \\ 11 \end{bmatrix}, \begin{bmatrix} 16 \\ 16 \end{bmatrix}, \cdots$$

If the initial departure times are chosen to be $\tau_1(1) = 1$ and $\tau_2(1) = 0$, the sequence is

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 5 \\ 4 \end{bmatrix}, \begin{bmatrix} 9 \\ 8 \end{bmatrix}, \begin{bmatrix} 13 \\ 12 \end{bmatrix}, \begin{bmatrix} 17 \\ 16 \end{bmatrix}, \cdots$$

In both cases the average departure interval is 4 units of time, however, in the second case the departure interval is constant, i.e., the system has a so-called 1-periodic behavior, while in the first case the system shows a 2-periodic behavior.

Eventually, the train company may consider to realize shorter (average) departure intervals. This could be achieved by using additional trains. For example the company provides a second train in the inner loop, e.g., initially on the track from station 1 to station 2. With respect to the timed event graph shown in Fig. 21.4, this means to add a second token in place p_2. In this case the firing times of transitions t_1 and t_2 as functions of π_i (Eq. 21.1) do not change, but the time instants when places p_i receive their tokens change to

$$\pi_1(k+1) = \tau_1(k) \tag{21.7}$$
$$\pi_2(k+2) = \tau_1(k) \tag{21.8}$$
$$\pi_3(k+1) = \tau_2(k) \tag{21.9}$$
$$\pi_4(k+1) = \tau_2(k). \tag{21.10}$$

Therefore the recursive equations for the firing times τ_1 and τ_2 change to

$$\tau_1(k+1) = \max\big(\tau_1(k)+2, \tau_2(k)+5\big) \tag{21.11}$$
$$\tau_2(k+2) = \max\big(\tau_1(k)+3, \tau_2(k+1)+3\big). \tag{21.12}$$

By introducing a new variable τ_3, with $\tau_3(k+1) := \tau_1(k)$, one may transform (21.11) and (21.12) into a system of first-order difference equations

$$\tau_1(k+1) = \max\big(\tau_1(k)+2, \tau_2(k)+5\big) \tag{21.13}$$
$$\tau_2(k+1) = \max\big(\tau_3(k)+3, \tau_2(k)+3\big) \tag{21.14}$$
$$\tau_3(k+1) = \tau_1(k). \tag{21.15}$$

Initializing this system with $\tau_1(1) = \tau_2(1) = \tau_3(1) = 0$, the following evolution can be determined:

$$\begin{bmatrix}0\\0\\0\end{bmatrix}, \begin{bmatrix}5\\3\\0\end{bmatrix}, \begin{bmatrix}8\\6\\5\end{bmatrix}, \begin{bmatrix}11\\9\\8\end{bmatrix}, \begin{bmatrix}14\\12\\11\end{bmatrix}, \cdots$$

After a short transient phase, trains depart from both stations at intervals of three units of time. Obviously, shorter intervals cannot be reached by additional trains in the inner loop of the system, as the outer loop at station 2 now represents the "bottleneck" of the system.

In this simple example, several phenomena have been encountered: 1-periodic solutions (for $\tau_1(1) = 1$ and $\tau_2(1) = 0$), 2-periodic solutions (for $\tau_1(1) = \tau_2(1) = 0$), and a transient phase (for the extended system). These phenomena (and more) can be conveniently analyzed and explained within the formal framework of idempotent semirings or dioids.

21.3 Dioid Algebraic Structures

An idempotent semiring (or dioid) is an algebraic structure based on two monoids (one of which is commutative). A structure $(M,\cdot,e)$ is a monoid if $\cdot$ is an internal law, associative and of which e is the identity element. If the law $\cdot$ is commutative, $(M,\cdot,e)$ is called a commutative monoid. Then, a dioid (or idempotent semiring) is a set, $\mathscr{D}$, endowed with two binary operations, addition $(\oplus)$ and multiplication $(\otimes)$, such that:

- $(\mathscr{D},\oplus,\varepsilon)$ is an idempotent commutative monoid, i.e., $\forall a \in \mathscr{D}, a \oplus a = a$
- $(\mathscr{D},\otimes,e)$ is a monoid
- $\otimes$ is distributive with respect to $\oplus$
- ε is absorbing for $\otimes$, i.e., $\forall a \in \mathscr{D}, \varepsilon \otimes a = a \otimes \varepsilon = \varepsilon$.

The identity element of addition (ε) is also called the zero element of the dioid, while the identity element of multiplication (e) is also called the unit element of the dioid. Often, the multiplication sign is omitted in written equations when they are unambiguous.

Due to the idempotency property, any dioid can be endowed with a natural (partial) order defined by $a \preceq b \Leftrightarrow a \oplus b = b$, i.e., the sum of two elements a and b is the least upper bound of a and b. Thus, any dioid forms a sub-semilattice. A dioid is said to be complete if it is closed for infinite sums, i.e., there exists the greatest element of $\mathscr{D}$ given by $\top = \bigoplus_{x \in \mathscr{D}} x$, and if $\otimes$ distributes over infinite sums. Formally, a complete dioid forms a complete lattice for which the greatest lower bound of a and b is denoted $a \wedge b$ [1].

Example 21.1. [Max-plus algebra] Probably the best known idempotent semiring is the so-called max-plus algebra, often denoted $\mathbb{Z}_{\max}$. It is defined over the set $\mathbb{Z} \cup \{-\infty\}$ and its binary operations are defined as follows:

- addition: $a \oplus b := \max(a,b)$
- multiplication: $a \otimes b := a + b$

and the zero and unit elements are $\varepsilon = -\infty$ and $e = 0$, respectively. Defining the operations over the set $\mathbb{Z} \cup \{-\infty,\infty\}$ defines a complete dioid (often denoted $\overline{\mathbb{Z}}_{\max}$), with the top element $\top = +\infty$. ■

Example 21.2. [Min-plus algebra] The set $\mathbb{Z} \cup \{-\infty,\infty\}$ endowed with addition defined as $a \oplus b := \min(a,b)$, and multiplication $(a \otimes b := a + b)$ forms a complete dioid also known as min-plus algebra, often denoted $\overline{\mathbb{Z}}_{\min}$. Its zero, unit, and top elements are $\varepsilon = \infty$, $e = 0$, and $\top = -\infty$, respectively. ■

Note that the (partial) order is a property of a given dioid. For two elements $a = 5$ and $b = 3$ the max-plus addition is defined as $a \oplus b = 5 \oplus 3 = 5$, which indicates that $5 \succeq 3$ in $\overline{\mathbb{Z}}_{\max}$. The same calculation can be done in min-plus algebra, i.e., a and b belonging to $\overline{\mathbb{Z}}_{\min}$. In this case we get $a \oplus b = 5 \oplus 3 = \min(5,3) = 3$ and, according to the definition of the natural order of a dioid, this means $5 \preceq 3$ in $\overline{\mathbb{Z}}_{\min}$.

As in classical algebra the binary operations can be extended to the matrix case in dioids. For the matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathscr{D}^{n\times m}$, and $\boldsymbol{C} \in \mathscr{D}^{m\times p}$ we get

$$\begin{aligned} \text{Addition:} \quad & [\boldsymbol{A} \oplus \boldsymbol{B}]_{ij} = [\boldsymbol{A}]_{ij} \oplus [\boldsymbol{B}]_{ij} \\ \text{Multiplication:} \quad & [\boldsymbol{A} \otimes \boldsymbol{C}]_{ij} = \bigoplus_{k=1}^{m} \Big([\boldsymbol{A}]_{ik} \otimes [\boldsymbol{C}]_{kj} \Big) . \end{aligned}$$

Note that max-plus algebra (and min-plus algebra) may also be defined on other sets, e.g., the set of real numbers. Depending on the set of definition, these idempotent semirings are denoted $\overline{\mathbb{R}}_{\max}$ or $\overline{\mathbb{R}}_{\min}$, respectively. An exhaustive description on idempotent semirings such as max-plus and min-plus algebra can be found e.g. in [1, 8, 11, 14].

21.4 Linear Dynamical Systems in Max-Plus Algebra

The usefulness of max-plus algebra becomes clear, when we take a second look at the previously introduced transportation network. Recall that the recursive equations for the earliest departure times in Fig. 21.4 are

$$\begin{aligned} \tau_1(k+1) &= \max\big(\tau_1(k)+2, \tau_2(k)+5\big) \\ \tau_2(k+1) &= \max\big(\tau_1(k)+3, \tau_2(k)+3\big). \end{aligned}$$

Due to the max-operation, these equations are nonlinear in conventional algebra. However, rewriting these equations in max-plus algebra with $\boldsymbol{x} := [\tau_1 \ \ \tau_2]^T$, results in the following linear system:

$$\begin{aligned} \boldsymbol{x}(k+1) &= \begin{bmatrix} 2 & 5 \\ 3 & 3 \end{bmatrix} \otimes \boldsymbol{x}(k) \\ &= \boldsymbol{A} \otimes \boldsymbol{x}(k). \end{aligned}$$

In general, it is possible to convert any timed event graph into a linear system in max-plus algebra, also called a "max-plus linear system". In max-plus algebra, the variable $x_i(k)$ is the earliest possible time instant that event x_i occurs for the k^{th} time. Therefore, max-plus algebraic functions $\boldsymbol{x}(k)$ are often called dater functions, giving every event a precise earliest date. Note that, besides max-plus algebra, it is also possible to convert any timed event graph into a min-plus linear system. Then, the min-plus variable $x_i(t)$ represents the maximal number of occurrences of event x_i up to time t. Min-plus algebraic functions are, therefore, often called counter functions.

As mentioned in Section 21.2, depending on the vector of initial firing times, a number of different phenomena have been observed: 1- and 2-periodic behaviors, with and without an initial transient phase. For many application scenarios as, e.g., transportation networks, a 1-periodic solution is desirable. It is therefore natural

to ask which initial firing vectors will indeed generate 1-periodic solutions and what the duration for one period is. In conventional algebra this means that the following equation shall be satisfied:

$$\tau_i(k+1) = \lambda + \tau_i(k), \quad \begin{matrix} k = 1,2,\ldots \\ i = 1,2,\ldots,n. \end{matrix}$$

Rewriting this requirement in max-plus algebra provides

$$x_i(k+1) = \lambda x_i(k), \quad \begin{matrix} k = 1,2,\ldots \\ i = 1,2,\ldots,n, \end{matrix}$$

or, equivalently,

$$\boldsymbol{x}(k+1) = \lambda \boldsymbol{x}(k), \quad k = 1,2,\ldots.$$

This amounts to solving the max-plus eigenproblem. If, for a given $\boldsymbol{A} \in \overline{\mathbb{Z}}_{\max}^{n\times n}$, there exists $\boldsymbol{\xi} \in \overline{\mathbb{Z}}_{\max}^{n}$ and $\lambda \in \mathbb{Z}$ such that

$$\boldsymbol{A}\boldsymbol{\xi} = \lambda \boldsymbol{\xi},$$

we call λ *eigenvalue* and $\boldsymbol{\xi}$ *eigenvector* of matrix A. If we choose the vector of initial firing times, $\boldsymbol{x}(1)$, as an eigenvector, we get

$$\boldsymbol{x}(2) = \boldsymbol{A}\boldsymbol{x}(1) = \lambda \boldsymbol{x}(1)$$

and therefore

$$\boldsymbol{x}(k) = \lambda^{(k-1)} \boldsymbol{x}(1), \quad k = 1,2,\ldots.$$

This is the desired 1-periodic behavior and the period length is the eigenvalue λ. Note that powers in max-plus algebra (and in dioids in general) are defined by $a^i = a \otimes a^{i-1}$, with $a^0 = e$.

To solve the max-plus eigenproblem, we need the notion of matrix (ir)reducibility, i.e, a matrix $\boldsymbol{A} \in \mathscr{D}^{n\times n}$ is called *reducible*, if there exists a permutation matrix $\boldsymbol{P}$, i.e., a square binary matrix that has exactly one 1-element in each column and row and zeros elsewhere, such that $\tilde{\boldsymbol{A}} = \boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^T$ is upper block-triangular. Otherwise, $\boldsymbol{A}$ is called *irreducible*. If $\boldsymbol{A} \in \mathscr{D}^{n\times n}$ is irreducible, there exists precisely one eigenvalue. It is given by

$$\lambda = \bigoplus_{j=1}^{n} \Big(tr\big(\boldsymbol{A}^j\big) \Big)^{1/j}, \tag{21.16}$$

where "trace" and the j^{th} root are defined as in conventional algebra, i.e., for any $\boldsymbol{B} \in \mathscr{D}^{n\times n}$,

$$tr(\boldsymbol{B}) = \bigoplus_{i=1}^{n} b_{ii}$$

and for any $\alpha \in \mathscr{D}$,

$$\left(\alpha^{1/j}\right)^j = \alpha.$$

Whereas an irreducible matrix $\boldsymbol{A} \in \mathscr{D}^{n\times n}$ has a unique eigenvalue λ, it may possess several distinct eigenvectors. We will not go into the details of how to compute the eigenvectors of matrices in dioids. There are several algorithms, e.g., Howard's algorithm and the power algorithm (see [14] for details), to determine the eigenvalue and the corresponding eigenvector(s).

Recalling the transportation network, we have determined the max-plus system matrix

$$\boldsymbol{A} = \begin{bmatrix} 2 & 5 \\ 3 & 3 \end{bmatrix} \quad \text{and one can check that} \quad \boldsymbol{A}^2 = \begin{bmatrix} 8 & 8 \\ 6 & 8 \end{bmatrix}.$$

According to (21.16), the eigenvalue of the system can be calculated by

$$\lambda = \bigoplus_{j=1}^{2} \left(tr\left(\boldsymbol{A}^j\right)\right)^{1/j} = 3^1 \oplus 8^{1/2} = 3 \oplus 4 = 4.$$

It turns out that $\boldsymbol{\xi} = [1 \ e]^T$ is an eigenvector of matrix $\boldsymbol{A}$, i.e., it satisfies $\boldsymbol{A} \otimes \boldsymbol{\xi} = \lambda \otimes \boldsymbol{\xi}$. Not surprisingly, this result confirms our observation of the transportation network. When the initial departure times are set to $\boldsymbol{x}(1) = [1 \ 0]^T = \boldsymbol{\xi}$, we obtain a 1-periodic schedule with a departure interval of $\lambda = 4$.

21.5 The 2-Dimensional Dioid $\mathscr{M}_{in}^{ax}[\![\gamma, \delta]\!]$

Max-plus algebra is, as shown in the previous section, suitable to model the behavior of timed event graphs. However, for more complex TEG, the corresponding representation in max-plus algebra becomes more complicated. For the transportation network with three trains in the inner loop, we had to introduce a new variable τ_3 to find the state model. Taking a look at another TEG (taken from [1]), this issue becomes even clearer. The linear dynamical system of Fig. 21.5 in max-plus algebra, with $x_i(k), u_j(k), y(k)$ being the earliest time instants that the transitions x_i, u_j, and y fire for the k^{th} time, is

$$\begin{aligned} x_1(k+1) &= 1u_1(k+1) \oplus 4x_2(k) \\ x_2(k+1) &= 5u_2(k) \oplus 3x_1(k+1) \\ x_3(k+1) &= 3x_1(k+1) \oplus 4x_2(k+1) \oplus 2x_3(k-1) \\ y(k+1) &= x_2(k) \oplus 2x_3(k+1). \end{aligned}$$

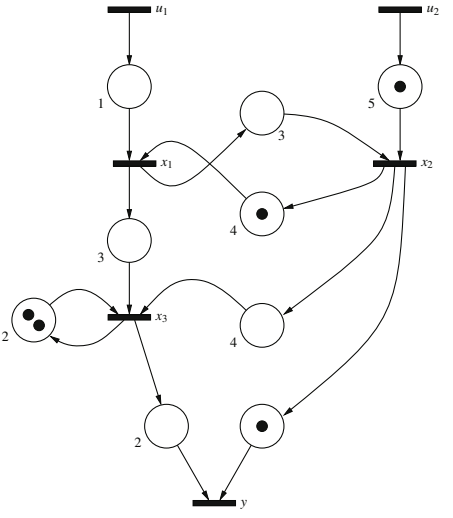

Fig. 21.5 Timed event graph (taken from [1])

Rewriting the linear system in matrix-vector form, we get

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A}_0\boldsymbol{x}(k+1) \oplus \boldsymbol{A}_1\boldsymbol{x}(k) \oplus \boldsymbol{A}_2\boldsymbol{x}(k-1) \oplus \boldsymbol{B}_0\boldsymbol{u}(k+1) \oplus \boldsymbol{B}_1\boldsymbol{u}(k) \\ y(k) &= \boldsymbol{C}_0\boldsymbol{x}(k) \oplus \boldsymbol{C}_1\boldsymbol{x}(k-1), \end{aligned} \tag{21.17}$$

where

$$\boldsymbol{A}_0 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ 3 & \varepsilon & \varepsilon \\ 3 & 4 & \varepsilon \end{bmatrix}, \quad \boldsymbol{A}_1 = \begin{bmatrix} \varepsilon & 4 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix}, \quad \boldsymbol{A}_2 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 2 \end{bmatrix},$$
$$\boldsymbol{B}_0 = \begin{bmatrix} 1 & \varepsilon \\ \varepsilon & \varepsilon \\ \varepsilon & \varepsilon \end{bmatrix}, \quad \boldsymbol{B}_1 = \begin{bmatrix} \varepsilon & \varepsilon \\ \varepsilon & 5 \\ \varepsilon & \varepsilon \end{bmatrix}, \quad \boldsymbol{C}_0 = \begin{bmatrix} \varepsilon & \varepsilon & 2 \end{bmatrix}, \quad \boldsymbol{C}_1 = \begin{bmatrix} \varepsilon & e & \varepsilon \end{bmatrix}.$$

In this example one observes that the equations are not in explicit form, and they are not first-order. Recursively inserting the equation into itself changes the system to

$$\begin{aligned} \boldsymbol{x}(k+1) =& \boldsymbol{A}_0\boldsymbol{x}(k+1) \oplus \boldsymbol{A}_1\boldsymbol{x}(k) \oplus \boldsymbol{A}_2\boldsymbol{x}(k-1) \oplus \boldsymbol{B}_0\boldsymbol{u}(k+1) \oplus \boldsymbol{B}_1\boldsymbol{u}(k) \\ =& \boldsymbol{A}_0 \underbrace{\big(\boldsymbol{A}_0\boldsymbol{x}(k+1) \oplus \boldsymbol{A}_1\boldsymbol{x}(k) \oplus \boldsymbol{A}_2\boldsymbol{x}(k-1) \oplus \boldsymbol{B}_0\boldsymbol{u}(k+1) \oplus \boldsymbol{B}_1\boldsymbol{u}(k)\big)}_{\boldsymbol{x}(k+1)} \oplus \\ & \boldsymbol{A}_1\boldsymbol{x}(k) \oplus \boldsymbol{A}_2\boldsymbol{x}(k-1) \oplus \boldsymbol{B}_0\boldsymbol{u}(k+1) \oplus \boldsymbol{B}_1\boldsymbol{u}(k) \\ =& \boldsymbol{A}_0^2\boldsymbol{x}(k+1) \oplus (\boldsymbol{I} \oplus \boldsymbol{A}_0)\big(\boldsymbol{A}_1\boldsymbol{x}(k) \oplus \boldsymbol{A}_2\boldsymbol{x}(k-1) \oplus \boldsymbol{B}_0\boldsymbol{u}(k+1) \oplus \boldsymbol{B}_1\boldsymbol{u}(k)\big) \\ =& \boldsymbol{A}_0^3\boldsymbol{x}(k+1) \oplus (\boldsymbol{I} \oplus \boldsymbol{A}_0 \oplus \boldsymbol{A}_0^2)\big(\boldsymbol{A}_1\boldsymbol{x}(k) \oplus \boldsymbol{A}_2\boldsymbol{x}(k-1) \oplus \boldsymbol{B}_0\boldsymbol{u}(k+1) \oplus \boldsymbol{B}_1\boldsymbol{u}(k)\big) \end{aligned}$$

Since matrix $\boldsymbol{A}_0$ is not cyclic, $\boldsymbol{A}_0^{\eta}$ contains only ε-entries for $\eta \geq n$. Therefore, the first term of the right-hand side is equal to zero, i.e.,

$$\begin{aligned}
\boldsymbol{x}(k+1) &= (\boldsymbol{I} \oplus \boldsymbol{A}_0 \oplus \boldsymbol{A}_0^2)\big(\boldsymbol{A}_1\boldsymbol{x}(k) \oplus \boldsymbol{A}_2\boldsymbol{x}(k-1) \oplus \boldsymbol{B}_0\boldsymbol{u}(k+1) \oplus \boldsymbol{B}_1\boldsymbol{u}(k)\big) \\
&= \begin{bmatrix} \varepsilon & 4 & \varepsilon \\ \varepsilon & 7 & \varepsilon \\ \varepsilon & 11 & \varepsilon \end{bmatrix} \boldsymbol{x}(k) \oplus \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 2 \end{bmatrix} \boldsymbol{x}(k-1) \\
&\quad \begin{bmatrix} 1 & \varepsilon \\ 4 & \varepsilon \\ 8 & \varepsilon \end{bmatrix} \boldsymbol{u}(k+1) \oplus \begin{bmatrix} \varepsilon & \varepsilon \\ \varepsilon & 5 \\ \varepsilon & 9 \end{bmatrix} \boldsymbol{u}(k), \\
y(k) &= \begin{bmatrix} \varepsilon & \varepsilon & 2 \end{bmatrix} \boldsymbol{x}(k) \oplus \begin{bmatrix} \varepsilon & e & \varepsilon \end{bmatrix} \boldsymbol{x}(k-1).
\end{aligned}$$

Then, we can transform this system into a first-order system by suitably augmenting the state space, e.g., by defining a new state vector $\tilde{\boldsymbol{x}}(k) = \begin{bmatrix} \boldsymbol{x}(k) & \boldsymbol{x}(k-1) & \boldsymbol{u}(k) \end{bmatrix}^T$. The resulting first-order system is

$$\begin{aligned}
\tilde{\boldsymbol{x}}(k+1) &= \tilde{\boldsymbol{A}}\tilde{\boldsymbol{x}}(k) \oplus \tilde{\boldsymbol{B}}\boldsymbol{u}(k+1) \\
y(k) &= \tilde{\boldsymbol{C}}\tilde{\boldsymbol{x}}(k),
\end{aligned}$$

where

$$\tilde{\boldsymbol{A}} = \begin{bmatrix} \varepsilon & 4 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 7 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 5 \\ \varepsilon & 11 & \varepsilon & \varepsilon & \varepsilon & 2 & \varepsilon & 9 \\ e & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}, \quad \tilde{\boldsymbol{B}} = \begin{bmatrix} 1 & \varepsilon \\ 4 & \varepsilon \\ 8 & \varepsilon \\ \varepsilon & \varepsilon \\ \varepsilon & \varepsilon \\ \varepsilon & \varepsilon \\ e & \varepsilon \\ \varepsilon & e \end{bmatrix}, \quad \tilde{\boldsymbol{C}} = \begin{bmatrix} \varepsilon & \varepsilon & 2 & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}. \tag{21.18}$$

Clearly, modeling even a simple TEG as given in Fig. 21.5 in max-plus algebra is not really efficient. However, applying the so-called γ-transform in max-plus algebra results in another idempotent semiring. The γ-transform is defined by $\boldsymbol{x}(\gamma) = \bigoplus_{k\in\mathbb{Z}} \boldsymbol{x}(k)\gamma^k$. The resulting dioid is denoted $\overline{\mathbb{Z}}_{\max}[\![\gamma]\!]$ and is the set of formal power series in one variable γ with coefficients in $\overline{\mathbb{Z}}_{\max}$ and exponents in $\mathbb{Z}$ [1]. Addition and multiplication of two formal power series s and s' are defined by

$$\begin{aligned}
\big(s \oplus s'\big)(k) &= s(k) \oplus s'(k), \\
\big(s \otimes s'\big)(k) &= \bigoplus_{i+j=k} s(i) \otimes s'(j).
\end{aligned}$$

The neutral element of addition is the series $\varepsilon(\gamma) = \bigoplus_{k\in\mathbb{Z}} \varepsilon\gamma^k$ and the neutral element of multiplication is the series $e(\gamma) = \bigoplus_{k\in\mathbb{N}} e\gamma^k$. Furthermore $\top(\gamma) = \bigoplus_{k\in\mathbb{Z}} \top\gamma^k$.

Applying the γ-transform to (21.17), we get

$$\gamma\boldsymbol{x}(\gamma) = \boldsymbol{A}_0\gamma\boldsymbol{x}(\gamma) \oplus \boldsymbol{A}_1\gamma^2\boldsymbol{x}(\gamma) \oplus \boldsymbol{A}_2\gamma^3\boldsymbol{x}(\gamma) \oplus \boldsymbol{B}_0\gamma\boldsymbol{u}(\gamma) \oplus \boldsymbol{B}_1\gamma^2\boldsymbol{u}(\gamma)$$

or equivalently

$$\begin{aligned}\boldsymbol{x}(\gamma) &= \underbrace{\left(\boldsymbol{A}_0 \oplus \gamma\boldsymbol{A}_1 \oplus \gamma^2\boldsymbol{A}_2\right)}_{\boldsymbol{A}(\gamma)}\boldsymbol{x}(\gamma) \oplus \underbrace{\left(\boldsymbol{B}_0 \oplus \gamma\boldsymbol{B}_1\right)}_{\boldsymbol{B}(\gamma)}\boldsymbol{u}(\gamma)\\ y(\gamma) &= \underbrace{\left(\boldsymbol{C}_0 \oplus \gamma\boldsymbol{C}_1\right)}_{\boldsymbol{C}(\gamma)}\boldsymbol{x}(\gamma)\end{aligned}$$

where

$$\boldsymbol{A}(\gamma) = \begin{bmatrix} \varepsilon & 4\gamma & \varepsilon \\ 3 & \varepsilon & \varepsilon \\ 3 & 4 & 2\gamma^2 \end{bmatrix}, \quad \boldsymbol{B}(\gamma) = \begin{bmatrix} 1 & \varepsilon \\ \varepsilon & 5\gamma \\ \varepsilon & \varepsilon \end{bmatrix}, \quad \boldsymbol{C}(\gamma) = \begin{bmatrix} \varepsilon & \gamma & 2 \end{bmatrix}. \tag{21.19}$$

Consequently, the TEG given in Fig. 21.5 can be modeled by (21.19), i.e., the γ-transformed system, instead of (21.18), i.e., the system one would obtain by augmenting the state space as described above. Hence, it may be more efficient to use the dioid $\overline{\mathbb{Z}}_{\max}[\![\gamma]\!]$ to model more complex TEG.

The system given in Fig. 21.5 can also be modeled in min-plus algebra (denoted $\overline{\mathbb{Z}}_{\min}$). In this case the state space has to be augmented as well, in order to obtain a first-order model. However, similar to the γ-transformation in max-plus algebra, there is the so-called δ-transformation in min-plus algebra. The transform constitutes an idempotent semiring denoted $\overline{\mathbb{Z}}_{\min}[\![\delta]\!]$. It is the set of formal power series in δ with coefficients in $\overline{\mathbb{Z}}_{\min}$ and exponents in $\mathbb{Z}$.

As mentioned before, it is possible to model any timed event graph in max-plus algebra as well as in min-plus algebra. Which dioid one uses is often based on the specific application which shall be modeled. If for example the model is supposed to be used in combination with a PLC (programmable logic controller) it may be favorable to model the system in min-plus algebra as the min-plus variable $\boldsymbol{x}(t)$ is dependent on time and a PLC also works with a specific cycle time.

It would, however, of course be preferable to combine the assets of max-plus and min-plus algebra. To do so, we first introduce a 2-dimensional dioid denoted $\mathbb{B}[\![\gamma,\delta]\!]$. It is the set of formal power series in two variables (γ,δ) with Boolean coefficients, i.e, $\mathbb{B} = \{\varepsilon, e\}$ and exponents in $\mathbb{Z}$ (see [1, 5, 10] for more information). Thus, a series in this dioid may be, for example, $s = \gamma^1\delta^1 \oplus \gamma^3\delta^2 \oplus \gamma^4\delta^5$. Graphically the series s can be represented as dots in the *event-time plane* (see Fig. 21.6). The interpretation of a monomial $\gamma^k\delta^t$ is that the $(k+1)^{\text{st}}$ occurrence of the corresponding event happens exactly at time t[1]. In terms of timed event graphs, however,

[1] Following the convention defined between Remarks 5.22 and 5.23 in [1, Section 5.4].

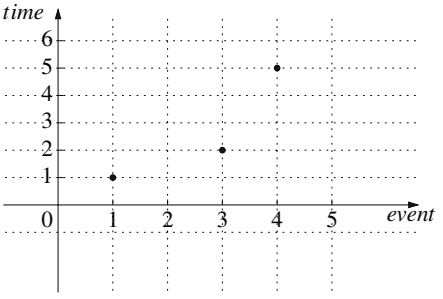

Fig. 21.6 Graphical representation of the series $s = \gamma^1\delta^1 \oplus \gamma^3\delta^2 \oplus \gamma^4\delta^5 \in \mathbb{B}[\![\gamma,\delta]\!]$

the following interpretation of $\gamma^k\delta^t$ would be more useful: *the* $(k+1)^{st}$ *occurrence of the event happens at time* t *at the earliest* or *at time* t*, the event has occurred at most* $(k+1)$ *times*. If, in this interpretation, we add two monomials $\gamma^k\delta^t$ and $\gamma^{k+\kappa}\delta^{t-\tau}$, $\kappa,\tau \in \mathbb{N}_0$, we clearly get $\gamma^k\delta^t$. In words, if an event can happen at time t at most $k+1$ times and it can happen at an earlier time $t-\tau$ at most $k+1+\kappa$ times, the resulting statement is that at time t it can happen at most $k+1$ times. We therefore identify all points $\gamma^k\delta^t$ and $\gamma^{k+\kappa}\delta^{t-\tau}$, $\kappa,\tau \in \mathbb{N}_0$, i.e., instead of a single point, we consider the "South-East cone" of this point. This establishes an equivalence relation and the resulting dioid of equivalence classes (quotient dioid) in $\mathbb{B}[\![\gamma,\delta]\!]$ is denoted $\mathscr{M}_{in}^{ax}[\![\gamma,\delta]\!]$ (see [1, 5, 10]). Note that this dioid admits only nondecreasing series and has the following properties:

$$\begin{aligned}\gamma^k\delta^t \oplus \gamma^l\delta^t &= \gamma^{\min(k,l)}\delta^t\\ \gamma^k\delta^t \oplus \gamma^k\delta^\tau &= \gamma^k\delta^{\max(t,\tau)}\\ \gamma^k\delta^t \otimes \gamma^l\delta^\tau &= \gamma^{(k+l)}\delta^{(t+\tau)}.\end{aligned}$$

The zero, unit, and top element of $\mathscr{M}_{in}^{ax}[\![\gamma,\delta]\!]$ are $\varepsilon = \gamma^{+\infty}\delta^{-\infty}$, $e = \gamma^0\delta^0$, and $\top = \gamma^{-\infty}\delta^{+\infty}$, respectively. As a consequence of this construction, e.g., the series $\tilde{s} = \gamma^1\delta^1 \oplus \gamma^3\delta^2 \oplus \gamma^4\delta^5 \oplus \gamma^5\delta^2$ is equivalent to the series $s = \gamma^1\delta^1 \oplus \gamma^3\delta^2 \oplus \gamma^4\delta^5$, which is shown in Fig. 21.7.

Using $\mathscr{M}_{in}^{ax}[\![\gamma,\delta]\!]$, we can immediately write down a dioid model for TEG. For example, the TEG in Fig. 21.5 can be represented by

$$\begin{aligned}x_1 &= \gamma^1\delta^4 x_2 \oplus \gamma^0\delta^1 u_1\\ x_2 &= \gamma^0\delta^3 x_1 \oplus \gamma^1\delta^5 u_2\\ x_3 &= \gamma^0\delta^3 x_1 \oplus \gamma^0\delta^4 x_2 \oplus \gamma^2\delta^2 x_3\\ y &= \gamma^1\delta^0 x_2 \oplus \gamma^0\delta^2 x_3\end{aligned}$$

or, more compactly,

$$\begin{aligned}\boldsymbol{x} &= \boldsymbol{A}\boldsymbol{x} \oplus \boldsymbol{B}\boldsymbol{u}\\ y &= \boldsymbol{C}\boldsymbol{x},\end{aligned}$$

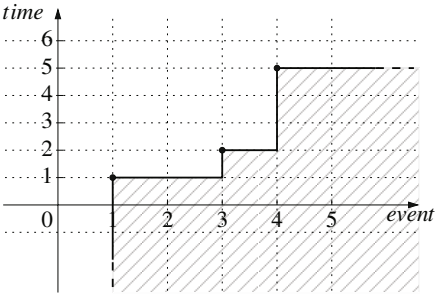

Fig. 21.7 Graphical representation of the series $s = \gamma^1\delta^1 \oplus \gamma^3\delta^2 \oplus \gamma^4\delta^5 \in \mathscr{M}_{in}^{ax}[\![\gamma,\delta]\!]$

where

$$\boldsymbol{A} = \begin{bmatrix} \varepsilon & \gamma^1\delta^4 & \varepsilon \\ \gamma^0\delta^3 & \varepsilon & \varepsilon \\ \gamma^0\delta^3 & \gamma^0\delta^4 & \gamma^2\delta^2 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} \gamma^0\delta^1 & \varepsilon \\ \varepsilon & \gamma^1\delta^5 \\ \varepsilon & \varepsilon \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} \varepsilon & \gamma^1\delta^0 & \gamma^0\delta^2 \end{bmatrix}.$$

Clearly, it is possible to model even very complex TEG in a compact form using the dioid $\mathscr{M}_{in}^{ax}[\![\gamma,\delta]\!]$.

21.6 High-Throughput Screening Systems

Among the vast variety of systems that can be modeled as timed event graphs is the operation of so-called *High-Throughput Screening Systems*. High-throughput screening (HTS) has become an important technology to rapidly test thousands of biochemical substances [13, 20]. In pharmaceutical industries, for example, HTS is often used for a first screening in the process of drug discovery. In general, high-throughput screening plants are fully automated systems containing a fixed set of devices performing liquid handling, storage, reading, plate handling, and incubation steps. All operations which have to be conducted to analyze one set of substances are combined in a so-called batch. The testing vessel carrying the biochemical substances in HTS systems is called microplate. It features a grid of up to 3456 wells. The number of wells is historically grown and represents a multiple of 96 reflecting the original microplate with 96 wells [18]. Several microplates may be included in a batch to convey reagents or waste material. While conducting a screening run, more than one batch may be present in the system at the same time, a single batch may pass the same machine more than once, a single batch may occupy two (or more) resources simultaneously, e.g., when being transferred from one resource to another, and there are minimal and maximal processing times defined by the user.

For better understanding we introduce a simple example of an HTS operation. One single batch of this example consists of three activities which are executed on

three different resources. The first activity, executed on R_1, represents the filling of some biochemical substance A into the wells of a microplate. After that, the microplate is moved to a second resource R_2, where the second activity is executed. This activity basically mixes substance A with some substance B. Substance B is provided by another activity executed on resource R_3. This resource operates independently of the other resources and may produce substance B even when resources R_1 and R_2 are not working. Due to hardware constraints the user sets the following minimal time durations:

- act_1 :
  - filling compound A into the wells of microplate: 7 units of time
  - post-processing after transfer of microplate to R_2: 1 unit of time
- act_2 :
  - pre-processing before transfer of microplate from R_1: 1 unit of time
  - waiting time before substance B can be added: 1 unit of time
  - mixing of substances: 12 units of time
- act_3 :
  - providing one heap of substance B: 5 units of time
  - post-processing after providing one heap of substance B: 1 unit of time
- The transfer processes are assumed to be possible in zero time.

The corresponding timed event graph is given in Fig. 21.8.

For real HTS systems, further activities, such as incubation steps or reading operations, would be executed on the compound AB. Screening runs of HTS plants may easily involve 150 resource allocations, i.e., activities, per batch. Since all relations

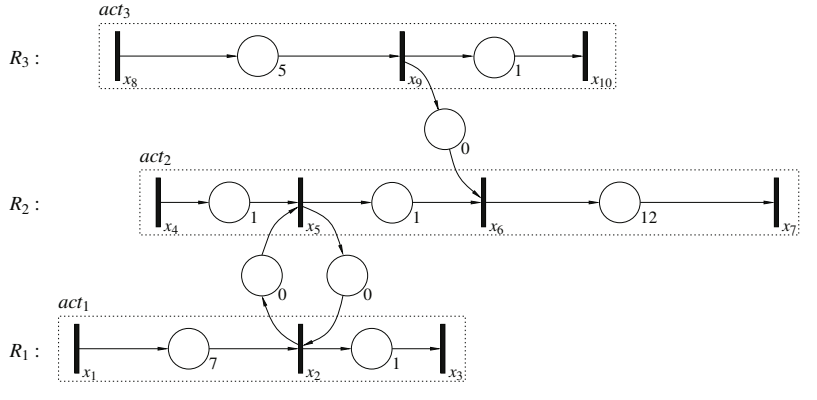

Fig. 21.8 TEG of a single batch of our example HTS operation

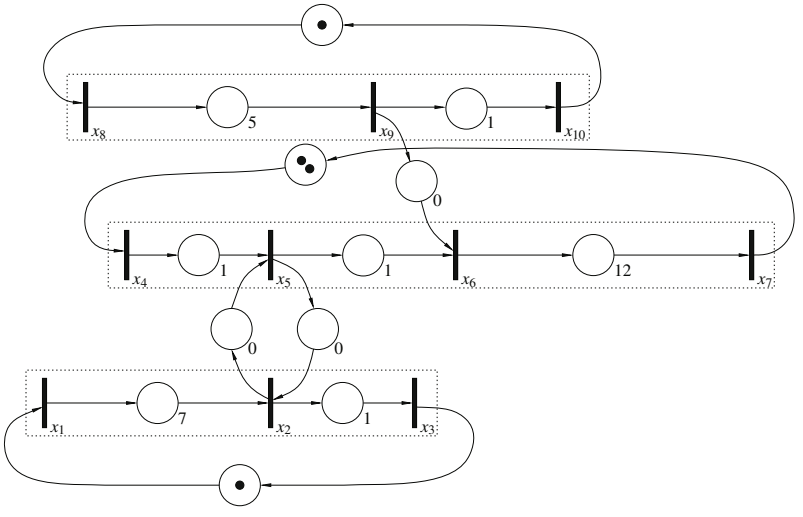

Fig. 21.9 TEG of a single batch and resource capacities

given in Fig. 21.8 belong to one single batch, there are no tokens in the timed event graph. However, additional constraints between events of different batches need to be taken into account. For example, all resources have a specific capacity. If a resource has a capacity of 1, an activity may only occupy this resource if the preceding activity (which may be of a different batch) has been finished. In our small example resources R_1 and R_3 have a capacity of 1 while resource R_2 has a capacity of 2, i.e., this resource may execute two activities at the same time. To model this, the TEG has to be extended by places with 1 and 2 initial tokens. The resulting TEG is shown in Fig. 21.9. Finally, the user has to evaluate which transitions he or she is able to control and what the output of the system is. For HTS plants, it is usually possible to control the starting events of every activity and the output is directly connected to the last transition of a single batch. Note that, depending on the specific system, the controllable transitions and the output transitions may be different. Without loss of generality, we fix all starting events of all activities of our example, i.e., x_1, x_4, and x_8, to be controllable and the output of our system shall be the last event of a single batch, i.e., x_7. Thus the timed event graph modeling the single batch of our HTS operation is extended according to Fig. 21.10. Then the TEG can be written as a $\mathcal{M}_{in}^{ax}[\![\gamma,\delta]\!]$-model:

$$\begin{aligned} \boldsymbol{x} &= \boldsymbol{A}\boldsymbol{x} \oplus \boldsymbol{B}\boldsymbol{u} \\ y &= \boldsymbol{C}\boldsymbol{x}, \end{aligned}$$

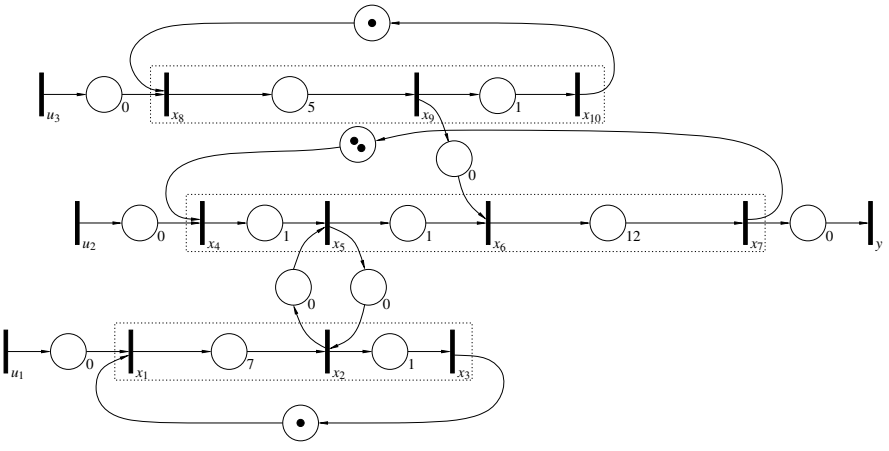

Fig. 21.10 TEG of a single batch, with resource capacities, and input and output transitions

with

$$\boldsymbol{A} = \begin{bmatrix} \varepsilon & \varepsilon & \gamma & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \delta^7 & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \delta & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \gamma^2 & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon & \delta & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta & \varepsilon & \varepsilon & \varepsilon & e \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^{12} & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \gamma \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta^5 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \delta & \varepsilon \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & e \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix},$$

$$\boldsymbol{C} = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & e & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}.$$

Clearly, the $\mathscr{M}_{in}^{ax}[\![\gamma, \delta]\!]$-model of the HTS operation is very compact. In the next chapter, it is explained how this model can be used to efficiently compute control laws for such systems.

21.7 Further Reading

In this chapter, the modeling and analysis of linear systems in a dioid framework have been presented. However, the chapter only provides a rough overview of this topic. A more exhaustive and mathematical presentation of dioids and systems in dioids can be found in [1, 2]. Many other works on max-plus algebra, dioids in general, and performance evaluation in idempotent semirings have been published,

e.g., [5, 8, 10, 11, 14]. Besides that, there are several software packages available to handle max-plus algebraic systems, e.g., the max-plus algebra toolbox for ScicosLab (`www.scicoslab.org`), or the C++ library *MinMaxGD* to manipulate periodic series in $\mathscr{M}_{in}^{ax}[\![\gamma,\delta]\!]$ [6]. For more information on high-throughput screening systems the reader is referred to [13, 18, 20]. However, please note, that HTS is only one possible application that has a linear representation in dioids. Other applications are, e.g., traffic systems, computer communication systems, production lines, and flows in networks. The reader is also invited to read the next chapter on control theory, developed for discrete-event systems in a dioid framework.

References

1. Baccelli, F., Cohen, G., Olsder, G.J., Quadrat, J.P.: Synchronization and Linearity–An Algebra for Discrete Event Systems. John Wiley and Sons (1992) Web Edition, `http://www-rocq.inria.fr/metalau/cohen/SED/SED1-book.html` (2012)
2. Blyth, T.S.: Lattices and Ordered Algebraic Structures. Springer (2005)
3. Bosscher, D., Polak, I., Vaandrager, F.: Verification of an Audio Control Protocol. In: Langmaack, H., de Roever, W.-P., Vytopil, J. (eds.) FTRTFT 1994 and ProCoS 1994. LNCS, vol. 863, pp. 170–192. Springer, Heidelberg (1994)
4. Cassandras, C.G., Lafortune, S., Olsder, G.J.: Introduction to the modelling, control and optimization of discrete event systems. In: Isidori, A. (ed.) Trends in Control–A European Perspective, pp. 217–291 (1995)
5. Cohen, G., Moller, P., Quadrat, J.-P., Viot, M.: Algebraic tools for the performance evaluation of discrete event systems. Proceedings of the IEEE 77(1), 39–58 (1989)
6. Cottenceau, B., Hardouin, L., Lhommeau, M., Boimond, J.L.: Data processing tool for calculation in dioid. In: Proc. 5th Int. Workshop on Discrete Event Systems, Ghent, Belgium (2000)
7. Cottenceau, B., Hardouin, L., Boimond, J.L., Ferrier, J.L.: Model reference control for timed event graphs in dioids. Automatica 37(9), 1451–1458 (2001)
8. Cuninghame-Green, R.A.: Minimax algebra. Lecture Notes in Economics and Mathematical Systems. Springer (1979)
9. Cuninghame-Green, R.A.: Minimax algebra and applications. Fuzzy Sets and Systems 41, 251–267 (1989)
10. Gaubert, S., Klimann, C.: Rational computation in dioid algebra and its application to performance evaluation of discrete event systems. In: Gérard, J., Lamnabhi-Lagarrigue, F. (eds.) Algebraic Computing in Control. LNCIS, vol. 165, pp. 241–252. Springer, Heidelberg (1991)
11. Gaubert, S.: Methods and Applications of (max,+) Linear Algebra. In: Reischuk, R., Morvan, M. (eds.) STACS 1997. LNCS, vol. 1200, pp. 261–282. Springer, Heidelberg (1997)
12. Gérard, J., Lamnabhi-Lagarrigue, F. (eds.): Algebraic Computing in Control. LNCIS, vol. 165. Springer (1991)
13. Harding, D., Banks, M., Fogarty, S., Binnie, A.: Development of an automated high-throughput screening system: a case history. Drug Discovery Today 2(9), 385–390 (1997)

14. Heidergott, B., Olsder, G.J., van der Woude, J.: Max Plus at Work–Modeling and Analysis of Synchronized Systems: A Course on Max-Plus Algebra and Its Applications. Princeton University Press, Princeton (2006)
15. Isidori, A. (ed.): Trends in Control – A European Perspective. Springer (1995)
16. Lhommeau, M., Hardouin, L., Cottenceau, B.: Disturbance decoupling of timed event graphs by output feedback controller. In: Proc. 6th Int. Workshop on Discrete Event Systems, Zaragoza, Spain (2002)
17. Maia, C.A., Hardouin, L., Santos-Mendes, R., Cottenceau, B.: Optimal closed-loop control of timed event graphs in dioids. IEEE Transactions on Automatic Control 48(12), 2284–2287 (2003)
18. Mayr, L.M., Fuerst, P.: The future of high-throughput screening. Journal of Biomolecular Screening 13(6), 443–448 (2008)
19. Reischuk, R., Morvan, M. (eds.): STACS 1997. LNCS, vol. 1200. Springer, Heidelberg (1997)
20. Rogers, M.V.: High-throughput screening. Drug Discovery Today 2(11), 503–504 (1997)