A Compositional Model for Multi-Rate Max-Plus Linear Systems

A Compositional Model for Multi-Rate Max-Plus Linear Systems ⋆

H. Elahi * **M. Geilen** * **T. Basten** *,**

* *Eindhoven University of Technology, Eindhoven, The Netherlands (e-mail: g.elahi, m.c.w.geilen, a.a.basten@tue.nl).*
** *ESI, TNO, Eindhoven, The Netherlands*

Abstract: The timing of discrete-event systems with synchronization is naturally modeled with canonical multi-rate max-plus linear equations. The main objectives of these models are to analyze and control the systems. As a system becomes more complex, determining its canonical model becomes more complicated. Moreover, these systems may change over time which demands the model to be recalculated. Motivated by the compositional structure of many systems, we propose operations to determine the canonical model for composing multi-rate max-plus linear systems. The operations allow efficient (re-)calculation of the canonical models from constituent canonical models. These models can be utilized to analyze and/or control complex systems using existing methods.

Keywords: Discrete-event systems, Max-plus linear systems, Compositional models

1. INTRODUCTION

Discrete-event systems (DESs) are extensively studied in literature. This research has concentrated on modeling, analysis and control of complex systems, ranging from multi-processing systems (e.g. Stuijk et al. (2007)) and telecommunication systems (e.g. Cruz (1991)) to transportation systems (e.g. Kersbergen et al. (2016)).

The phenomenon of synchronization in DESs is a nonlinear characteristic in classic system theory that can be modeled as a linear aspect in max-plus algebra (see Baccelli et al. (1992)). These DESs with synchronization are referred to as Max-Plus Linear Systems (MPLSs). Max-plus algebra provides an opportunity to apply some of the classical linear system approaches for such systems, such as model predictive control for MPLSs (De Schutter and Van den Boom (2001)). Moreover, it facilitates to evaluate performance properties of a system such as throughput (Ghamarian et al. (2006)).

Finding a canonical max-plus linear model of a complex system is a challenging task. Furthermore, a system may dynamically change from one configuration to another. Consider a software update for an autonomous vehicle as an example. The performance of this system must be guaranteed. For instance, a short delay in the response time of this system may reduce the reliability of this system and lead to catastrophic circumstances. Hence, prior to this update, the performance of the system must be carefully analyzed and verified. To evaluate and verify the performance of the system, the canonical-form representation of this system is determined. This system has multiple heterogenous applications mapped onto a heterogenous distributed shared platform and a wide verity of sensors and actuators. The applications and resources have multiple configurations that vary over time. Moreover, depending on the dynamics of the environment, an application might be added or removed. Therefore, these systems are sophisticated and (re-)calculation of their model is consequently complicated. Nonetheless, these systems are likely to be created from simpler components, such as image filtering applications, object detection applications and video tracking applications. These components have their own canonical models and interact with each other through their inputs and outputs.

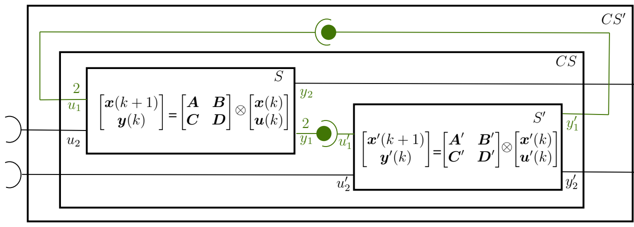


Fig. 1. A system constructed from components S and S'

A closed-form symbolic formulation reduces the complexity of modeling complex systems, built from simpler MPLSs, in canonical form. In particular, for a system with multiple configurations, it is convenient to determine the canonical-form representation of its actual configuration based on the canonical model of its components without flattening. Therefore, we propose a compositional model of DESs described by canonical max-plus linear equations.

As a running example, Fig. 1 depicts a system with two components S and S', characterized by max-plus-linear equations. S has two input ports, u_1 and u_2, represented by two-pronged forks, and two output ports, y_1 and y_2, depicted by lollipops. Similarly, u'_1 and u'_2 are input ports

⋆ This research was supported by the Electronic Components and Systems for European Leadership (ECSEL) Joint Undertaking under grant number H2020- ECSEL-2017-2-783162 through the FitOptiVis project (FitOptiVis (2019)).

of S', and S' has two output ports y'_1 and y'_2. Input and output ports communicate discrete-time signals. These signals capture the production times of events, not the data values that are exchanged. In every execution of the system, called an *iteration*, the system consumes a fixed number of samples from each input port and produces a fixed number of samples on each output. In figures, sample rates (samples per iteration) greater than one are annotated above ports. In Fig. 1, annotation 2 above port y_1 gives the sample rate of y_1. Hence, these systems are called Multi-rate Max-Plus Linear Systems (M^2PLSs).

Fig. 1 illustrates two M^2PLSs S and S' producing samples for each other. Output y_1 of S is connected to input u'_1 of S', and output y'_1 of S' is connected to input u_1 of S. Our goal is to compute the canonical model of the total system from the canonical models of S and S'. First, the canonical-form representation of CS, which is a composite model of S and S' with a connection from y_1 to u'_1, is determined. From the connection from y_1 to u'_1 in Fig. 1, it follows that two samples are produced on y_1 in every iteration of S, while in every iteration of S' only one sample is read from u'_1. To handle these unbalanced rates, first, we synchronize the rates. In this example, a model of S' for two iterations is determined, which consumes two input samples in every execution. After rate synchronization, the model of CS can be determined through substitution. Then, the canonical model of CS' is determined as a composite model of CS after adding a connection from y'_1 to u_1.

Our proposed method perceives each system as a black box with its canonical model. The determined canonical model abstracting the composite system can be used to analyze or control that composite system. Our compositional model is an algebraic method that composes M^2PLSs characterized in the canonical form using two operations. The first operation synchronizes the sample rates of the systems to be composed. The second operation captures a connection from an output port to an input port. New connections may introduce deadlocks. A deadlock is a state in which a group of components waits for synchronization among them in a cyclic way. Our method checks such dependencies in the canonical max-plus linear model of the system before adding a connection.

2. RELATED WORK

Widespread applications of max-plus linear system theory have been investigated in the literature. The targets of this research can be divided into three main purposes: (I) modeling, (II) performance analysis and (III) control and optimization. This paper focuses on modeling of composite max-plus linear systems for the purpose of analysis and/or control of complex DESs. Although no research concentrates on the composition of max-plus linear systems as a generic problem, research often applies max-plus algebra to model, analyze or control a specific compositional system. For instance, a worst-case response time analysis for parallel compositions of synchronous systems was proposed by Aguado et al. (2017). In the following two paragraphs, two compositional methods for modeling, analysis and control of specific DESs, namely dataflow systems and manufacturing systems, are reviewed.

The max-plus semantics of dataflow models is used to analyze complex hierarchical dataflow models. Skelin and Geilen (2017) propose a method to evaluate throughput of hierarchical synchronous dataflow models. Their method is an extension to the max-plus semantics of synchronous dataflow models that facilitates throughput analysis of hierarchical models without flattening them. To design and predict behavior of complex applications of which the behavior changes modes with a deterministic, periodic pattern, a compositional dataflow model was suggested by Alizadeh Ara et al. (2018). They utilize the max-plus semantics of dataflow models to model and efficiently analyze the behavior of complex applications. Although both of these methods facilitate the analysis of composite systems, their methods are only useful in their specific domain and do not treat components as black boxes.

To analyze and control manufacturing systems created from simpler systems, usually in a serial form, some research focuses on the canonical max-plus linear model of these systems. For instance, a model for the composition of manufacturing systems has been proposed by Imaev and Judd (2008). Their hierarchical model uses canonical max-plus equations to calculate performance properties of a system such as machine utilization and work in process. However, their paper considers only serial composition of manufacturing systems, where a sequence of jobs passes from one system to another. Seleim and ElMaraghy (2014) introduce a max-plus model of manufacturing flow lines. They suggest a model for merging lines as well as for serial systems. Their method facilitates analysis for reconfiguring flow lines. However, their method is presented informally and is specific for their case study.

None of the aforementioned research addresses arbitrary compositions of *MPLS*s even in their specific domains, whereas we propose an algebraic method to find the canonical model of any arbitrary composition of M^2PLSs as a generic problem.

3. PRELIMINARIES

This section introduces the necessary mathematical preliminaries for this paper. For more detailed information, see Baccelli et al. (1992) and Heidergott et al. (2005).

3.1 Max-Plus Algebra

For $a, b \in \mathbb{R}_{max} = \mathbb{R} \cup \{-\infty\}$, the $\oplus$ and $\otimes$ operations are defined as $a \oplus b \triangleq max(a, b)$ and $a \otimes b \triangleq a + b$. As in linear algebra, the set $\mathbb{R}_{max}$ with operations can be extended to vectors in $\mathbb{R}^n_{max}$ and matrices in $\mathbb{R}^{n\times m}_{max}$, where $n, m \in \mathbb{N}$. For $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n\times m}_{max}$, $\boldsymbol{A} \oplus \boldsymbol{B}$ is defined by $[\boldsymbol{A} \oplus \boldsymbol{B}]_{(i,j)} = \boldsymbol{A}_{(i,j)} \oplus \boldsymbol{B}_{(i,j)}$, where (i, j) denotes the element of row i and column j of the matrix. To multiply two matrices $\boldsymbol{A} \in \mathbb{R}^{n\times m}_{max}$ and $\boldsymbol{B} \in \mathbb{R}^{m\times p}_{max}$, $[\boldsymbol{A} \otimes \boldsymbol{B}]_{(i,j)} \triangleq \bigoplus_{k=1}^{m} \boldsymbol{A}_{(i,k)} \otimes \boldsymbol{B}_{(k,j)}$. The b^{th} power of $\boldsymbol{A} \in \mathbb{R}^{n\times n}_{max}$ is $\boldsymbol{A}^b = \underbrace{\boldsymbol{A} \otimes \ldots \otimes \boldsymbol{A}}_{b}$.

3.2 Max-Plus Linear Systems

An important subclass of discrete-event systems for which only synchronization and delay are the key aspects of exe-

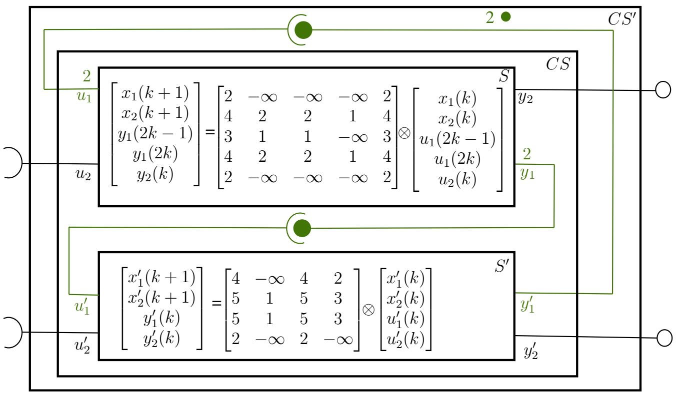


Fig. 2. The running example, a composite system is created from two M^2PLSs.

cution is called max-plus linear systems. Synchronization means that an operation waits until all preceding operations have been completed. This behavior can be modeled by the $\oplus$ operation in max-plus algebra. Delay means that an operation executes in a fixed amount of time, which is modeled by operation $\otimes$ in max-plus algebra. The characteristic equations of max-plus linear systems are described in canonical form as follows (see Baccelli et al. (1992) and Heidergott et al. (2005)).

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

where $\boldsymbol{x}(k)$, $\boldsymbol{u}(k)$ and $\boldsymbol{y}(k)$ are discrete-time signals that represent the production times of states, inputs, and outputs respectively. $\boldsymbol{A}$ is called the state matrix, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$ are called input, output and feed-through matrices, respectively.

4. PROBLEM FORMULATION

Let MRS be a set of M^2PLSs to be composed, e.g. S and $S' \in MRS$. Let U and Y be the sets of inputs and outputs of the MPLSs. For instance, in Fig. 2, $u_1, u_2, u'_1, u'_2 \in U$. $Ip : MRS \to \mathbb{P}(U)$ specifies the set of input identifiers of each system (e.g. $Ip(S) = \{u_1, u_2\}$), where $\mathbb{P}(U)$ denotes the power set of U. Similarly, $Op : MRS \to \mathbb{P}(Y)$ specifies the set of output identifiers of each system. These identifiers are annotated below the input and output ports in the graphical representation. Input and output ports communicate discrete-time signals with elements from $\mathbb{R}_{max}$. Systems are repeatedly executed during which they read a fixed number of samples from each input and produce a fixed number of samples on each output. Such an execution is called an iteration, which is captured by one iteration of the canonical equations of the M^2PLS.

In one iteration, example system S reads two samples from input u_1 and one sample from input u_2. It produces two samples on output y_1 and one sample on output y_2. *Sample rate* $Sr : U \cup Y \to \mathbb{N}$ gives for each port the number of samples consumed or produced in one iteration of a system (e.g. $Sr(y_1) = 2$). Rates are denoted by a number above the ports. For simplicity, rates of one are not shown. A vector of samples produced on $y_i \in Op(S)$ (consumed from $u_i \in Ip(S)$) for any $S \in MRS$, in the k^{th} iteration of S, is indicated by $\boldsymbol{y}_i(k)$ ($\boldsymbol{u}_i(k)$). A vector of all samples produced (consumed) in the k^{th} iteration of S is denoted by $\boldsymbol{y}(k)$ ($\boldsymbol{u}(k)$). For S in Fig. 2, $\boldsymbol{y}_1(k) = [y_1(2k-1)\ y_1(2k)]^T$ and S produces output vector $\boldsymbol{y}(k) = [y_1(2k-1)\ y_1(2k)\ y_2(k)]^T$ during its iteration k.

A *connection* is an injective partial function $OI : Y \hookrightarrow U$ that specifies connections from outputs to inputs (e.g., $OI(y_1) = u'_1$). To prevent a mutual sample dependency (when output and input samples of two M^2PLSs depend on one another), a number of samples may be initially available. The number of *initial samples* on connections is specified by $Is : OI \to \mathbb{N}_0$ (e.g., $Is(y'_1, u_1) = 2$). This is shown in the graphical representation with a dot alongside the number of available samples. At first, these available initial samples are read and, as a result, produced samples are consumed with delay.

The problem we address in this paper is the following. Given the canonical models of two M^2PLSs S and S', a connection function OI and an initial sample function Is, we want to obtain the canonical-form M^2PLS representation of the composition of the two systems, where the matrices of the composite M^2PLS are expressed directly in terms of the matrices $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ and $\boldsymbol{A}'$, $\boldsymbol{B}'$, $\boldsymbol{C}'$, and $\boldsymbol{D}'$ of the constituent M^2PLSs. Note that this two-component composition enables the composition of an arbitrary number of components.

A composite system has a canonical model only if the system is deadlock-free and consistent. A system is deadlock-free when there is no mutual dependency between its samples.

To define consistency of a system, we generalize the definitions of repetition vector and consistency for dataflow graphs of Lee (1991). When a system is executed iteratively, the numbers of produced and consumed samples on the connections must be equal in each iteration. Let $r_i \in \mathbb{N}$ be the number of times component S_i is repeated in each iteration of the system. The following defines consistency as the equality of the production and consumption sample rates on all connections under r_i.

$$\forall (y_j, u_i) \in OI \quad : r_j Sr(y_j) = r_i Sr(u_i), \quad (2)$$

where y_j and u_i denote an output of component S_j and an input of component S_i, respectively.

Definition 1. [Consistency] A composite system built from components $S_1, \ldots, S_n$ is consistent if and only if there is a vector $\boldsymbol{r} = [r_1 \ldots r_n]^T$ with strictly positive elements that satisfies (2). The smallest such solution $\boldsymbol{r}$ is called the repetition vector.

In case no solution for the equations in (2) exists, the composite system is inconsistent and consequently has no canonical model. Inconsistency implies that a system may deadlock or that the delay between producing a sample and consuming that same sample grows without a bound. Inconsistent composite systems are therefore not meaningful. In the remainder, we only consider consistent compositions. Consistency is not sufficient for deadlock-freedom though. Also a consistent composite system may deadlock because of insufficient initial samples on its connections. A sufficient condition for deadlock-freedom of the composite system is checked in the derivation of the canonical-form model of the composition.

5. A COMPOSITIONAL MODEL OF M^2PLSs

This section presents an algebraic method for finding the canonical model of M^2PLS compositions. Our method determines symbolically the canonical max-plus linear model of a composite system based on the canonical models of simpler systems from which it is constructed.

Fig. 3 (d) shows a system created from our two example M^2PLSs, S and S'. Output y_1 of S is connected to input u'_1 of S' and y'_1 of S' is connected to u_1 of S. S produces m samples on y_1 and consumes m samples from u_1, while m' samples are consumed from u'_1 and produced on y'_1 in system S' (where $m = 2$ and $m' = 1$ in the earlier concrete examples). Composite system CS can only be consistent if $Sr(y_1) = Sr(u'_1)$ and $Sr(y'_1) = Sr(u_1)$ in the composite system. So if $m \neq m'$, we need to synchronize these rates before composition. Constituents of a composite system are synchronized and aggregated into a synchronized-rate model using operation $\mathcal{RS}$ specified in *Definition* 2 (Fig. 3 (a)) in Section 5.1. Next, the connection from y_1 to u'_1 is realized and the canonical model of CS is determined using operation $\mathcal{IO}$ of *Definition* 3 (Fig. 3 (b)) in Section 5.2. The second connection is realized by applying again operation $\mathcal{IO}$ to the model of CS (See Fig. 3 (c)).

5.1 Rate synchronization

In Fig. 3 (d), connection $oi = (y_1, u'_1) \in OI$ between S and S' is such that $Sr(y_1) = m$ and $Sr(u'_1) = m'$. This implies that every iteration of S produces m samples on oi; every iteration of S' consumes m' samples from oi. The models of S and S' can be synchronized using the repetition vector of the composite system (*Definition* 1). In the example, $\boldsymbol{r} = [r\ r']^T$ with $r = \frac{m'}{gcd(m,m')}$, $r' = \frac{m}{gcd(m,m')}$ and $gcd(m, m')$ the greatest common divisor of m and m'. This results in the sample rate $rm = r'm'$ for both y_1 and u'_1. Also the rates of the second connection are synchronized in this way.

We need a model of an M^2PLS after a given number of iterations. The values of the states after every p iterations ($\boldsymbol{x}(kp+1) \in \mathbb{R}^n_{max}$) and every single output sample produced during these p iterations can be determined from the values of the states before these p iterations ($\boldsymbol{x}((k-1)p+1)$) and all the input samples consumed during these p iterations. Substituting $k+1$ for k in (1) gives:

$$\begin{aligned}\boldsymbol{x}(k+2) &= \boldsymbol{A}\otimes\boldsymbol{x}(k+1) \oplus \boldsymbol{B}\otimes\boldsymbol{u}(k+1) \\ \boldsymbol{y}(k+1) &= \boldsymbol{C}\otimes\boldsymbol{x}(k+1) \oplus \boldsymbol{D}\otimes\boldsymbol{u}(k+1).\end{aligned} \tag{3}$$

Substituting $\boldsymbol{x}(k+1)$ from (1) into (3) and keeping $\boldsymbol{y}(k)$ from (1) yields the following:

$$\begin{bmatrix}\boldsymbol{x}(k+2)\\ \boldsymbol{y}(k)\\ \boldsymbol{y}(k+1)\end{bmatrix} = \begin{bmatrix}\boldsymbol{A}\otimes\boldsymbol{A} & \boldsymbol{A}\otimes\boldsymbol{B} & \boldsymbol{B}\\ \boldsymbol{C} & \boldsymbol{D} & -\infty\\ \boldsymbol{C}\otimes\boldsymbol{A} & \boldsymbol{C}\otimes\boldsymbol{B} & \boldsymbol{D}\end{bmatrix} \otimes \begin{bmatrix}\boldsymbol{x}(k)\\ \boldsymbol{u}(k)\\ \boldsymbol{u}(k+1)\end{bmatrix} \tag{4}$$

Equation (4) shows how the states of a system after two iterations $\boldsymbol{x}(k+2)$ and the output samples of the first and second iterations ($\boldsymbol{y}(k)$ and $\boldsymbol{y}(k+1)$) can be obtained from states $\boldsymbol{x}(k)$ and inputs $\boldsymbol{u}(k)$ and $\boldsymbol{u}(k+1)$ of the system during these two iterations. This method can be extended for p iterations of a system. The states after p iterations ($\boldsymbol{x}(k+p)$) and outputs from the first iteration ($\boldsymbol{y}(k)$) up to the p^{th} iteration ($\boldsymbol{y}(k+p-1)$) can be determined from states $\boldsymbol{x}(k)$ and inputs $\boldsymbol{u}(k)$ up to $\boldsymbol{u}(k+p-1)$, (5).

$$\begin{bmatrix}\boldsymbol{x}(k+p)\\ \boldsymbol{y}(k)\\ \vdots\\ \boldsymbol{y}(k+p-1)\end{bmatrix} = \boldsymbol{M}\otimes\begin{bmatrix}\boldsymbol{x}(k)\\ \boldsymbol{u}(k)\\ \vdots\\ \boldsymbol{u}(k+p-1)\end{bmatrix} \tag{5}$$

$$\text{with } \boldsymbol{M} = \begin{bmatrix}\boldsymbol{A}^p & \boldsymbol{A}^{p-1}\otimes\boldsymbol{B} & \cdots & \boldsymbol{B}\\ \boldsymbol{C} & \boldsymbol{D} & \cdots & -\infty\\ \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{C}\otimes\boldsymbol{A}^{p-1} & \boldsymbol{C}\otimes\boldsymbol{A}^{p-2}\otimes\boldsymbol{B} & \cdots & \boldsymbol{D}\end{bmatrix}.$$

To calculate matrices after every p iterations, substituting $p(k-1)+1$ for k in (5) yields:

$$\begin{bmatrix}\boldsymbol{x}^{*p}(k+1)\\ \boldsymbol{y}^{*p}(k)\end{bmatrix} = \begin{bmatrix}\boldsymbol{A}^{*p} & \boldsymbol{B}^{*p}\\ \boldsymbol{C}^{*p} & \boldsymbol{D}^{*p}\end{bmatrix}\otimes\begin{bmatrix}\boldsymbol{x}^{*p}(k)\\ \boldsymbol{u}^{*p}(k)\end{bmatrix} \tag{6}$$

where

$$\boldsymbol{x}^{*p}(k+1) = \boldsymbol{x}(pk+1),$$

$$\boldsymbol{y}^{*p}(k) = \begin{bmatrix}\boldsymbol{y}(p(k-1)+1)\\ \boldsymbol{y}(p(k-1)+2)\\ \vdots\\ \boldsymbol{y}(pk)\end{bmatrix},\ \boldsymbol{u}^{*p}(k) = \begin{bmatrix}\boldsymbol{u}(p(k-1)+1)\\ \boldsymbol{u}(p(k-1)+2)\\ \vdots\\ \boldsymbol{u}(pk)\end{bmatrix},$$

$$\boldsymbol{A}^{*p} = \boldsymbol{A}^p,\ \boldsymbol{B}^{*p} = \left[\boldsymbol{A}^{p-1}\otimes\boldsymbol{B}\ \boldsymbol{A}^{p-2}\otimes\boldsymbol{B}\ \ldots\ \boldsymbol{A}\otimes\boldsymbol{B}\ \boldsymbol{B}\right],$$

$$\boldsymbol{C}^{*p} = \begin{bmatrix}\boldsymbol{C}\\ \boldsymbol{C}\otimes\boldsymbol{A}\\ \vdots\\ \boldsymbol{C}\otimes\boldsymbol{A}^{p-1}\end{bmatrix} \text{ and}$$

$$\boldsymbol{D}^{*p} = \begin{bmatrix}\boldsymbol{D} & -\infty & -\infty & \cdots & -\infty\\ \boldsymbol{C}\otimes\boldsymbol{B} & \boldsymbol{D} & -\infty & \cdots & -\infty\\ \boldsymbol{C}\otimes\boldsymbol{A}\otimes\boldsymbol{B} & \boldsymbol{C}\otimes\boldsymbol{B} & \boldsymbol{D} & \cdots & -\infty\\ \vdots & \ddots & \ddots & \ddots & \vdots\\ \boldsymbol{C}\otimes\boldsymbol{A}^{p-2}\otimes\boldsymbol{B} & \cdots & \boldsymbol{C}\otimes\boldsymbol{A}\otimes\boldsymbol{B} & \boldsymbol{C}\otimes\boldsymbol{B} & \boldsymbol{D}\end{bmatrix}.$$

Definition 2. [Synchronized-rate model] $AS = \mathcal{RS}(S, S', \boldsymbol{r})$ is an operation taking the canonical models of two M^2PLSs S and S', and repetition vector $\boldsymbol{r} = [r\ r']^T$ to equalize the production and consumption sample rates (on the required connections between them). It returns the canonical model of an M^2PLS that is an aggregated model of S^{*r} and $S'^{*r'}$. S^{*r} and $S'^{*r'}$ are the canonical models of S after every r iterations and S' after every r' iterations and can be determined from (6). Equation (7) shows the max-plus linear equations of AS.

$$\begin{bmatrix}\boldsymbol{x}^{\mathcal{RS}}(k+1)\\ \boldsymbol{y}^{\mathcal{RS}}(k)\end{bmatrix} = \begin{bmatrix}\boldsymbol{A}^{\mathcal{RS}} & \boldsymbol{B}^{\mathcal{RS}}\\ \boldsymbol{C}^{\mathcal{RS}} & \boldsymbol{D}^{\mathcal{RS}}\end{bmatrix}\otimes\begin{bmatrix}\boldsymbol{x}^{\mathcal{RS}}(k)\\ \boldsymbol{u}^{\mathcal{RS}}(k)\end{bmatrix}, \tag{7}$$

where

$$\boldsymbol{x}^{\mathcal{RS}}(k) = \begin{bmatrix}\boldsymbol{x}^{*r}(k)\\ \boldsymbol{x}'^{*r'}(k)\end{bmatrix}, \boldsymbol{y}^{\mathcal{RS}}(k) = \begin{bmatrix}\boldsymbol{y}^{*r}(k)\\ \boldsymbol{y}'^{*r'}(k)\end{bmatrix},$$

$$\boldsymbol{u}^{\mathcal{RS}}(k) = \left[\boldsymbol{u}^{*r}(k)\boldsymbol{u}'^{*r'}(k)\right]^T,$$

$$\boldsymbol{A}^{\mathcal{RS}} = \begin{bmatrix}\boldsymbol{A}^{*r} & -\infty\\ -\infty & \boldsymbol{A}'^{*r'}\end{bmatrix}, \boldsymbol{B}^{\mathcal{RS}} = \begin{bmatrix}\boldsymbol{B}^{*r} & -\infty\\ -\infty & \boldsymbol{B}'^{*r'}\end{bmatrix},$$

$$\boldsymbol{C}^{\mathcal{RS}} = \begin{bmatrix}\boldsymbol{C}^{*r} & -\infty\\ -\infty & \boldsymbol{C}'^{*r'}\end{bmatrix} \text{ and } \boldsymbol{D}^{\mathcal{RS}} = \begin{bmatrix}\boldsymbol{D}^{*r} & -\infty\\ -\infty & \boldsymbol{D}'^{*r'}\end{bmatrix}.$$

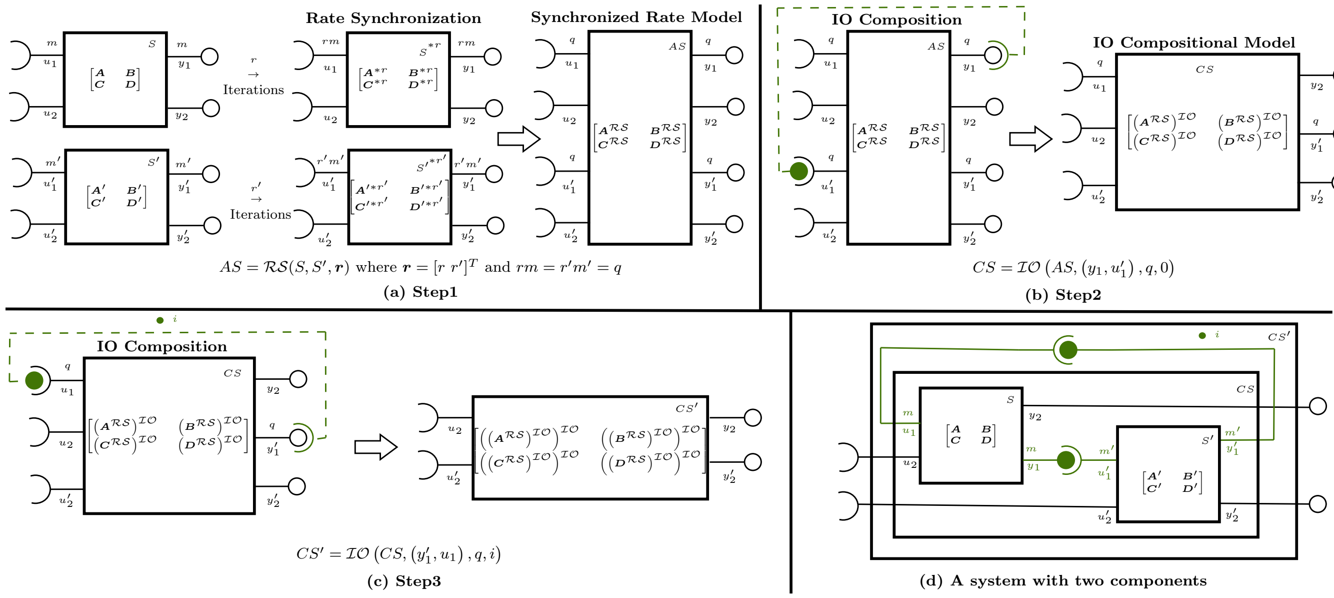


Fig. 3. Composing two M^2PLSs S and S' by adding two connections stepwise

$\boldsymbol{A}^{*r}$, $\boldsymbol{B}^{*r}$, $\boldsymbol{C}^{*r}$ and $\boldsymbol{D}^{*r}$ are matrices of S after every r iterations, while $\boldsymbol{A}'^{*r'}$, $\boldsymbol{B}'^{*r'}$, $\boldsymbol{C}'^{*r'}$ and $\boldsymbol{D}'^{*r'}$ are matrices of S' after every r' iterations.

To synchronize S and S' of the running example of Fig. 2, $AS = \mathcal{RS}(S, S', [1\ 2]^T)$ is computed (see Fig. 3 (a)). First, the model of S' for two iterations is calculated from (6):

$$\boldsymbol{A}'^{*2} = \boldsymbol{A}' \otimes \boldsymbol{A}' = \begin{bmatrix} 4 & -\infty \\ 5 & 1 \end{bmatrix} \otimes \begin{bmatrix} 4 & -\infty \\ 5 & 1 \end{bmatrix} = \begin{bmatrix} 8 & -\infty \\ 9 & 2 \end{bmatrix},$$

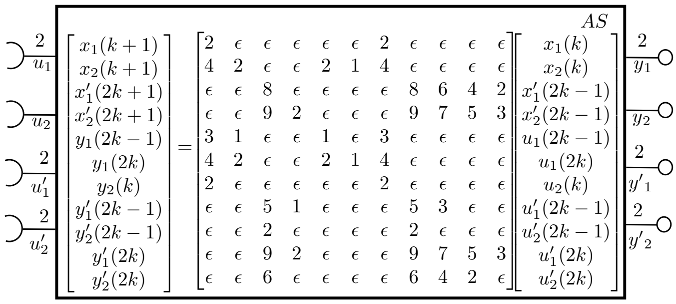
$$\boldsymbol{B}'^{*2} = \begin{bmatrix} \boldsymbol{A}' \otimes \boldsymbol{B}' & \boldsymbol{B}' \end{bmatrix} = \begin{bmatrix} 8 & 6 & 4 & 2 \\ 9 & 7 & 5 & 3 \end{bmatrix},$$

$$\boldsymbol{C}'^{*2} = \begin{bmatrix} \boldsymbol{C}' \\ \boldsymbol{C}' \otimes \boldsymbol{A}' \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 5 & 1 \\ 2 & -\infty \end{bmatrix} \\ \begin{bmatrix} 5 & 1 \\ 2 & -\infty \end{bmatrix} \otimes \begin{bmatrix} 4 & -\infty \\ 5 & 1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 5 & 1 \\ 2 & -\infty \\ 9 & 2 \\ 6 & -\infty \end{bmatrix},$$

$$\text{and } \boldsymbol{D}'^{*2} = \begin{bmatrix} \boldsymbol{D}' & -\infty \\ \boldsymbol{C}' \otimes \boldsymbol{B}' & \boldsymbol{D}' \end{bmatrix} = \begin{bmatrix} 5 & 3 & -\infty & -\infty \\ 2 & -\infty & -\infty & -\infty \\ 9 & 7 & 5 & 3 \\ 6 & 4 & 2 & -\infty \end{bmatrix}.$$

The synchronized-rate model in canonical form of this example is depicted in Fig. 4, where $\epsilon = -\infty$. For the purpose of illustration, the top left 4×4 matrix in this figure denotes $\boldsymbol{A}^{\mathcal{RS}}$,

$$\boldsymbol{A}^{\mathcal{RS}} = \begin{bmatrix} 2 & -\infty & -\infty & -\infty \\ 4 & 2 & -\infty & -\infty \\ -\infty & -\infty & 8 & -\infty \\ -\infty & -\infty & 9 & 2 \end{bmatrix}.$$



$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_1'(2k+1) \\ x_2'(2k+1) \\ y_1(2k-1) \\ y_1(2k) \\ y_2(k) \\ y_1'(2k-1) \\ y_2'(2k-1) \\ y_1'(2k) \\ y_2'(2k) \end{bmatrix} = \begin{bmatrix} 2 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 2 & \epsilon & \epsilon & \epsilon & \epsilon \\ 4 & 2 & \epsilon & \epsilon & 2 & 1 & 4 & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & 8 & \epsilon & \epsilon & \epsilon & \epsilon & 8 & 6 & 4 & 2 \\ \epsilon & \epsilon & 9 & 2 & \epsilon & \epsilon & \epsilon & 9 & 7 & 5 & 3 \\ 3 & 1 & \epsilon & \epsilon & 1 & \epsilon & 3 & \epsilon & \epsilon & \epsilon & \epsilon \\ 4 & 2 & \epsilon & \epsilon & 2 & 1 & 4 & \epsilon & \epsilon & \epsilon & \epsilon \\ 2 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 2 & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & 5 & 1 & \epsilon & \epsilon & \epsilon & 5 & 3 & \epsilon & \epsilon \\ \epsilon & \epsilon & 2 & \epsilon & \epsilon & \epsilon & \epsilon & 2 & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & 9 & 2 & \epsilon & \epsilon & \epsilon & 9 & 7 & 5 & 3 \\ \epsilon & \epsilon & 6 & \epsilon & \epsilon & \epsilon & \epsilon & 6 & 4 & 2 & \epsilon \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_1'(2k-1) \\ x_2'(2k-1) \\ u_1(2k-1) \\ u_1(2k) \\ u_2(k) \\ u_1'(2k-1) \\ u_2'(2k-1) \\ u_1'(2k) \\ u_2'(2k) \end{bmatrix}$$

Fig. 4. The synchronized-rate model of the example

5.2 IO Connection

After synchronizing and aggregating the canonical models of two systems S and S', using operation $\mathcal{RS}$ of *Definition* 2, the connection $io = (y_1, u_1')$ is added to the canonical model of AS (Fig. 3 (b)). To formulate adding connections in general, consider that a connection $oi = (y_1, u_1) \in OI$ with $Sr(y_1) = Sr(u_1) = m$ is added to the model of $S \in MRS$, such that $Is(oi) = i$, with $y_1 \in Op(S)$ and $u_1 \in Ip(S)$. To determine the canonical model considering this connection, the inputs and outputs of S are divided into two groups. Let u_1 and y_1 be the input and the output to be connected. Therefore, $\boldsymbol{y}_1(k)$ is the vector of produced samples on this connection oi, while $\boldsymbol{u}_1(k)$ indicates the vector of consumed samples from oi. Let the vector of the rest of the samples produced on the other connections be $\boldsymbol{y}_2(k)$. Likewise, let the vector of the rest of the input samples be $\boldsymbol{u}_2(k)$. According to this notation, the system equations can be rewritten as follows:

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A} \otimes \boldsymbol{x}(k) \oplus \boldsymbol{B}_1 \otimes \boldsymbol{u}_1(k) \oplus \boldsymbol{B}_2 \otimes \boldsymbol{u}_2(k) \\ \boldsymbol{y}_1(k) &= \boldsymbol{C}_1 \otimes \boldsymbol{x}(k) \oplus \boldsymbol{D}_{1,1} \otimes \boldsymbol{u}_1(k) \oplus \boldsymbol{D}_{1,2} \otimes \boldsymbol{u}_2(k) \\ \boldsymbol{y}_2(k) &= \boldsymbol{C}_2 \otimes \boldsymbol{x}(k) \oplus \boldsymbol{D}_{2,1} \otimes \boldsymbol{u}_1(k) \oplus \boldsymbol{D}_{2,2} \otimes \boldsymbol{u}_2(k) \end{aligned} \quad (8)$$

After adding connection oi, y_1 and u_1 are no longer an output and input of the composite component. Therefore, the values of these vectors should be captured in terms of other signals and eliminated from the canonical model of the composite system. For this purpose, in general, we follow two strategies: (I) eliminating those produced samples consumed within an iteration and (II) saving those produced samples that are not consumed in the same iteration in auxiliary vectors called augmented states. The problem is divided into two cases: (I) the number of initial samples i on the connection is less than the u_1 and y_1 sample rates m and (II) $i \geq m$.

In case $0 \leq i < m$, some of the produced outputs are consumed within the same iteration. For the system to be deadlock-free, no (earlier) outputs should depend on some of the (later) inputs. Then it is sufficient if the open loop matrix looks like:

$$\begin{bmatrix} \boldsymbol{x}(k+1) \\ \boldsymbol{y}_{1a}(k) \\ \boldsymbol{y}_{1b}(k) \\ \boldsymbol{y}_2(k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B}_{1a} & \boldsymbol{B}_{1b} & \boldsymbol{B}_2 \\ \boldsymbol{C}_{1a} & \boldsymbol{D}_{1a,1a} & -\boldsymbol{\infty} & \boldsymbol{D}_{1a,2} \\ \boldsymbol{C}_{1b} & \boldsymbol{D}_{1b,1a} & \boldsymbol{D}_{1b,1b} & \boldsymbol{D}_{1b,2} \\ \boldsymbol{C}_2 & \boldsymbol{D}_{2,1a} & \boldsymbol{D}_{2,1b} & \boldsymbol{D}_{2,2} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{u}_{1a}(k) \\ \boldsymbol{u}_{1b}(k) \\ \boldsymbol{u}_2(k) \end{bmatrix}, \tag{9}$$

where y_1 is split into y_{1a} and y_{1b} and u_1 into u_{1a} and u_{1b} such that y_{1a} is the output that is produced and immediately consumed by input u_{1b} within an iteration. Input u_{1a} initially consumes the initial samples on the connection. After the first iteration, u_{1a} reads the later samples produced by y_{1b} in the previous iteration. To ensure deadlock-freedom, it suffices if submatrix $\boldsymbol{D}_{1a,1b} = -\boldsymbol{\infty}_{(m-i)\times(m-i)}$, which is an $(m-i)\times(m-i)$ matrix with $-\infty$ entries. $\boldsymbol{D}_{1a,1b}$ describes the dependency of $\boldsymbol{y}_{1a}(k)$ on $\boldsymbol{u}_{1b}(k)$. It is not feasible to consider $\boldsymbol{u}_{1b}(k) = \boldsymbol{y}_{1a}(k)$ because these two are waiting for one another, at the same time. This mutual dependency between $\boldsymbol{u}_{1b}(k)$ and $\boldsymbol{y}_{1a}(k)$ introduces deadlock in the system. From (9), $\boldsymbol{y}_{1a}(k) = [\boldsymbol{C}_{1a}\ \boldsymbol{D}_{1a,1a}\ \boldsymbol{D}_{1a,2}] \otimes [\boldsymbol{x}(k)\ \boldsymbol{u}_{1a}(k)\ \boldsymbol{u}_2(k)]^T$ and as a result of $\boldsymbol{u}_{1b}(k) = \boldsymbol{y}_{1a}(k)$, the following equations are true:

$$\begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{u}_{1a}(k) \\ \boldsymbol{u}_{1b}(k) \\ \boldsymbol{u}_2(k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} & -\boldsymbol{\infty} & -\boldsymbol{\infty} \\ -\boldsymbol{\infty} & \boldsymbol{I} & -\boldsymbol{\infty} \\ \boldsymbol{C}_{1a} & \boldsymbol{D}_{1a,1a} & \boldsymbol{D}_{1a,2} \\ -\boldsymbol{\infty} & -\boldsymbol{\infty} & \boldsymbol{I} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{u}_{1a}(k) \\ \boldsymbol{u}_2(k) \end{bmatrix}, \tag{10}$$

where $\boldsymbol{I}$ is a max-plus identity matrix with zeros on the main diagonal and $-\infty$ elsewhere. Substituting the right-hand side of (10) for $[\boldsymbol{x}(k)\ \boldsymbol{u}_{1a}(k)\ \boldsymbol{u}_{1b}(k)\ \boldsymbol{u}_2(k)]^T$ in (9) yields:

$$\begin{bmatrix} \boldsymbol{x}(k+1) \\ \boldsymbol{y}_{1a}(k) \\ \boldsymbol{y}_{1b}(k) \\ \boldsymbol{y}_2(k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B}_{1a} & \boldsymbol{B}_{1b} & \boldsymbol{B}_2 \\ \boldsymbol{C}_{1a} & \boldsymbol{D}_{1a,1a} & -\boldsymbol{\infty} & \boldsymbol{D}_{1a,2} \\ \boldsymbol{C}_{1b} & \boldsymbol{D}_{1b,1a} & \boldsymbol{D}_{1b,1b} & \boldsymbol{D}_{1b,2} \\ \boldsymbol{C}_2 & \boldsymbol{D}_{2,1a} & \boldsymbol{D}_{2,1b} & \boldsymbol{D}_{2,2} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{I} & -\boldsymbol{\infty} & -\boldsymbol{\infty} \\ -\boldsymbol{\infty} & \boldsymbol{I} & -\boldsymbol{\infty} \\ \boldsymbol{C}_{1a} & \boldsymbol{D}_{1a,1a} & \boldsymbol{D}_{1a,2} \\ -\boldsymbol{\infty} & -\boldsymbol{\infty} & \boldsymbol{I} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{u}_{1a}(k) \\ \boldsymbol{u}_2(k) \end{bmatrix} =$$

$$\begin{bmatrix} \boldsymbol{A} \oplus \boldsymbol{B}_{1b} \otimes \boldsymbol{C}_{1a} & \boldsymbol{B}_{1a} \oplus \boldsymbol{B}_{1b} \otimes \boldsymbol{D}_{1a,1a} & \boldsymbol{B}_2 \oplus \boldsymbol{B}_{1b} \otimes \boldsymbol{D}_{1a,2} \\ \boldsymbol{C}_{1a} & \boldsymbol{D}_{1a,1a} & \boldsymbol{D}_{1a,2} \\ \boldsymbol{C}_{1b} \oplus \boldsymbol{D}_{1b,1b} \otimes \boldsymbol{C}_{1a} & \boldsymbol{D}_{1b,1a} \oplus \boldsymbol{D}_{1b,1b} \otimes \boldsymbol{D}_{1a,1a} & \boldsymbol{D}_{1b,2} \oplus \boldsymbol{D}_{1b,1b} \otimes \boldsymbol{D}_{1a,2} \\ \boldsymbol{C}_2 \oplus \boldsymbol{D}_{2,1b} \otimes \boldsymbol{C}_{1a} & \boldsymbol{D}_{2,1a} \oplus \boldsymbol{D}_{2,1b} \otimes \boldsymbol{D}_{1a,1a} & \boldsymbol{D}_{2,2} \oplus \boldsymbol{D}_{2,1b} \otimes \boldsymbol{D}_{1a,2} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{u}_{1a}(k) \\ \boldsymbol{u}_2(k) \end{bmatrix} \tag{11}$$

Given that the value of $\boldsymbol{y}_{1a}(k)$ is already incorporated in $\boldsymbol{x}(k+1)$, $\boldsymbol{y}_{1b}(k)$ and $\boldsymbol{y}_2(k)$ in (11), its corresponding (the second) row is removed from (11), which results in:

$$\begin{bmatrix} \boldsymbol{x}(k+1) \\ \boldsymbol{y}_{1b}(k) \\ \boldsymbol{y}_2(k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} \oplus \boldsymbol{B}_{1b} \otimes \boldsymbol{C}_{1a} & \boldsymbol{B}_{1a} \oplus \boldsymbol{B}_{1b} \otimes \boldsymbol{D}_{1a,1a} & \boldsymbol{B}_2 \oplus \boldsymbol{B}_{1b} \otimes \boldsymbol{D}_{1a,2} \\ \boldsymbol{C}_{1b} \oplus \boldsymbol{D}_{1b,1b} \otimes \boldsymbol{C}_{1a} & \boldsymbol{D}_{1b,1a} \oplus \boldsymbol{D}_{1b,1b} \otimes \boldsymbol{D}_{1a,1a} & \boldsymbol{D}_{1b,2} \oplus \boldsymbol{D}_{1b,1b} \otimes \boldsymbol{D}_{1a,2} \\ \boldsymbol{C}_2 \oplus \boldsymbol{D}_{2,1b} \otimes \boldsymbol{C}_{1a} & \boldsymbol{D}_{2,1a} \oplus \boldsymbol{D}_{2,1b} \otimes \boldsymbol{D}_{1a,1a} & \boldsymbol{D}_{2,2} \oplus \boldsymbol{D}_{2,1b} \otimes \boldsymbol{D}_{1a,2} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{u}_{1a}(k) \\ \boldsymbol{u}_2(k) \end{bmatrix} \tag{12}$$

To capture the values of the newly added initial samples on connection oi and y_{1b}, let $\boldsymbol{\chi}(k+1) \in \mathbb{R}^{i}_{max}$ be a new augmented state vector indicating the values of the remaining samples after iteration k. In iteration k, u_{1a} reads the values of remaining samples from iteration $k-1$ which is $\boldsymbol{\chi}(k)$. After iteration k, vector $\boldsymbol{y}_{1b}(k)$ produces samples which are not consumed in iteration k; thus, $\boldsymbol{\chi}(k+1) = \boldsymbol{y}_{1b}(k)$. Consequently, $\boldsymbol{\chi}(k+1)$ and $\boldsymbol{\chi}(k)$ can be substituted for $\boldsymbol{y}_{1b}(k)$ and $\boldsymbol{u}_{1a}(k)$ in (12), respectively. These substitutions yield the following results describing CS, which is the canonical model of S after adding connection $oi = (y_1, u_1)$.

$$\begin{bmatrix} \boldsymbol{x}^{\mathcal{IO}}(k+1) \\ \boldsymbol{y}^{\mathcal{IO}}(k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}^{\mathcal{IO}} & \boldsymbol{B}^{\mathcal{IO}} \\ \boldsymbol{C}^{\mathcal{IO}} & \boldsymbol{D}^{\mathcal{IO}} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}^{\mathcal{IO}}(k) \\ \boldsymbol{u}^{\mathcal{IO}}(k) \end{bmatrix}, \tag{13}$$

where

$$\boldsymbol{x}^{\mathcal{IO}}(k) = [\boldsymbol{x}(k)\ \boldsymbol{\chi}(k)]^T,\ \boldsymbol{u}^{\mathcal{IO}}(k) = \boldsymbol{u}_2(k),\ \boldsymbol{y}^{\mathcal{IO}}(k) = \boldsymbol{y}_2(k),$$

$$\boldsymbol{A}^{\mathcal{IO}} = \begin{bmatrix} \boldsymbol{A} \oplus \boldsymbol{B}_{1b} \otimes \boldsymbol{C}_{1a} & \boldsymbol{B}_{1a} \oplus \boldsymbol{B}_{1b} \otimes \boldsymbol{D}_{1a,1a} \\ \boldsymbol{C}_{1b} \oplus \boldsymbol{D}_{1b,1b} \otimes \boldsymbol{C}_{1a} & \boldsymbol{D}_{1b,1a} \oplus \boldsymbol{D}_{1b,1b} \otimes \boldsymbol{D}_{1a,1a} \end{bmatrix},$$

$$\boldsymbol{B}^{\mathcal{IO}} = \begin{bmatrix} \boldsymbol{B}_2 \oplus \boldsymbol{B}_{1b} \otimes \boldsymbol{D}_{1a,2} \\ \boldsymbol{D}_{1b,2} \oplus \boldsymbol{D}_{1b,1b} \otimes \boldsymbol{D}_{1a,2} \end{bmatrix},$$

$$\boldsymbol{C}^{\mathcal{IO}} = [\boldsymbol{C}_2 \oplus \boldsymbol{D}_{2,1b} \otimes \boldsymbol{C}_{1a}\ \ \boldsymbol{D}_{2,1a} \oplus \boldsymbol{D}_{2,1b} \otimes \boldsymbol{D}_{1a,1a}]$$

$$\text{and } \boldsymbol{D}^{\mathcal{IO}} = \boldsymbol{D}_{2,2} \oplus \boldsymbol{D}_{2,1b} \otimes \boldsymbol{D}_{1a,2}.$$

In case $i \geq m$, no samples are consumed in the same iteration in which they are produced. Therefore, the system after adding the connection is deadlock-free. Thus, in contrast to the previous case, the condition $\boldsymbol{D}_{1a,1b} = -\boldsymbol{\infty}$ is not necessary. The canonical model of S is described as follows:

$$\begin{bmatrix} \boldsymbol{x}(k+1) \\ \boldsymbol{y}_{1a}(k) \\ \boldsymbol{y}_{1b}(k) \\ \boldsymbol{y}_2(k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B}_{1a} & \boldsymbol{B}_{1b} & \boldsymbol{B}_2 \\ \boldsymbol{C}_{1a} & \boldsymbol{D}_{1a,1a} & \boldsymbol{D}_{1a,1b} & \boldsymbol{D}_{1a,2} \\ \boldsymbol{C}_{1b} & \boldsymbol{D}_{1b,1a} & \boldsymbol{D}_{1b,1b} & \boldsymbol{D}_{1b,2} \\ \boldsymbol{C}_2 & \boldsymbol{D}_{2,1a} & \boldsymbol{D}_{2,1b} & \boldsymbol{D}_{2,2} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{u}_{1a}(k) \\ \boldsymbol{u}_{1b}(k) \\ \boldsymbol{u}_2(k) \end{bmatrix}. \tag{14}$$

As a result of i initial samples on connection oi, the first part of the produced samples $\boldsymbol{y}_{1a}(k) \in \mathbb{R}^{(m-mod(i,m))}_{max}$ on connection oi is read in iteration $k + \lfloor \frac{i}{m} \rfloor$ by u_{1b}, where $mod(i,m)$ is the remainder of the division of i by m. The rest of the produced samples in iteration k ($\boldsymbol{y}_{1b}(k) \in \mathbb{R}^{mod(i,m)}_{max}$) is consumed by $\boldsymbol{u}_{1a}(k + \lceil \frac{i}{m} \rceil)$. Therefore, $\boldsymbol{y}_{1a}(k) = \boldsymbol{u}_{1b}(k + \lfloor \frac{i}{m} \rfloor)$ and $\boldsymbol{y}_{1b}(k) = \boldsymbol{u}_{1a}(k + \lceil \frac{i}{m} \rceil)$. To be able to formulate and include $\boldsymbol{y}_{1a}(k) = \boldsymbol{u}_{1b}(k + \lfloor \frac{i}{m} \rfloor)$ in the canonical-form representation, $\lfloor \frac{i}{m} \rfloor$ new augmented state vectors called $\boldsymbol{\chi}_{1a}(k)$, $\boldsymbol{\chi}_{2a}(k)$, $\ldots$, $\boldsymbol{\chi}_{(\lfloor \frac{i}{m} \rfloor - 1)a}(k)$, $\boldsymbol{\chi}_{\lfloor \frac{i}{m} \rfloor a}(k) \in \mathbb{R}^{(m-mod(i,m))}_{max}$ are defined. Likewise, $\lceil \frac{i}{m} \rceil$ new augmented state vectors $\boldsymbol{\chi}_{1b}(k)$, $\ldots$, $\boldsymbol{\chi}_{(\lceil \frac{i}{m} \rceil - 1)b}(k)$, $\boldsymbol{\chi}_{\lceil \frac{i}{m} \rceil b}(k) \in \mathbb{R}^{mod(i,m)}_{max}$ are added to address $\boldsymbol{y}_{1b}(k) = \boldsymbol{u}_{1a}(k + \lceil \frac{i}{m} \rceil)$. To capture the delay between the production and consumption of the samples, the relations between vectors are defined as follows:

$$\begin{bmatrix} \boldsymbol{\chi}_{1a}(k+1) \\ \boldsymbol{\chi}_{2a}(k+1) \\ \vdots \\ \boldsymbol{\chi}_{(\lfloor \frac{i}{m} \rfloor - 1)a}(k+1) \\ \boldsymbol{\chi}_{\lfloor \frac{i}{m} \rfloor a}(k+1) \\ \boldsymbol{u}_{1b}(k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_{1a}(k) \\ \boldsymbol{\chi}_{1a}(k) \\ \vdots \\ \boldsymbol{\chi}_{(\lfloor \frac{i}{m} \rfloor - 2)a}(k) \\ \boldsymbol{\chi}_{(\lfloor \frac{i}{m} \rfloor - 1)a}(k) \\ \boldsymbol{\chi}_{\lfloor \frac{i}{m} \rfloor a}(k) \end{bmatrix} \text{ and} \tag{15}$$

$$\begin{bmatrix} \boldsymbol{\chi}_{1b}(k+1) \\ \boldsymbol{\chi}_{2b}(k+1) \\ \vdots \\ \boldsymbol{\chi}_{(\lceil \frac{i}{m} \rceil -1)b}(k+1) \\ \boldsymbol{\chi}_{\lceil \frac{i}{m} \rceil b}(k+1) \\ \boldsymbol{u}_{1a}(k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_{1b}(k) \\ \boldsymbol{\chi}_{1b}(k) \\ \vdots \\ \boldsymbol{\chi}_{(\lceil \frac{i}{m} \rceil -2)b}(k) \\ \boldsymbol{\chi}_{(\lceil \frac{i}{m} \rceil -1)b}(k) \\ \boldsymbol{\chi}_{\lceil \frac{i}{m} \rceil b}(k) \end{bmatrix}. \tag{16}$$

Let's define $\boldsymbol{\chi}_{a1}(k) = \left[\boldsymbol{\chi}_{1a}(k)\ \boldsymbol{\chi}_{2a}(k)\ \cdots\ \boldsymbol{\chi}_{(\lfloor \frac{i}{m} \rfloor -1)a}(k)\right]^T$ and $\boldsymbol{\chi}_{a}(k) = \left[\boldsymbol{\chi}_{a1}(k)\ \boldsymbol{\chi}_{\lfloor \frac{i}{m} \rfloor a}(k)\right]^T$; likewise, $\boldsymbol{\chi}_{b1}(k) = \left[\boldsymbol{\chi}_{1b}(k)\ \cdots\ \boldsymbol{\chi}_{(\lceil \frac{i}{m} \rceil -1)b}(k)\right]^T$, $\boldsymbol{\chi}_{b}(k) = \left[\boldsymbol{\chi}_{b1}(k)\ \boldsymbol{\chi}_{\lceil \frac{i}{m} \rceil b}(k)\right]^T$. Thus, from (15) and (16) the following equations are true:

$$\boldsymbol{\chi}_a(k+1) = \begin{bmatrix} \boldsymbol{y}_{1a}(k) \\ \boldsymbol{\chi}_{a1}(k) \end{bmatrix} \text{ and } \boldsymbol{\chi}_b(k+1) = \begin{bmatrix} \boldsymbol{y}_{1b}(k) \\ \boldsymbol{\chi}_{b1}(k) \end{bmatrix}. \tag{17}$$

Substituting $\boldsymbol{y}_{1a}(k)$ from (14) results in:

$$\boldsymbol{\chi}_a(k+1) = \begin{bmatrix} \boldsymbol{C}_{1a} & \boldsymbol{\epsilon} & \boldsymbol{D}_{1a,1b} & \boldsymbol{\epsilon} & \boldsymbol{D}_{1a,1a} & \boldsymbol{D}_{1a,2} \\ \boldsymbol{\epsilon} & \boldsymbol{I} & \boldsymbol{\epsilon} & \boldsymbol{\epsilon} & \boldsymbol{\epsilon} & \boldsymbol{\epsilon} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{\chi}_{a1}(k) \\ \boldsymbol{u}_{1b}(k) \\ \boldsymbol{\chi}_{b1}(k) \\ \boldsymbol{u}_{1a}(k) \\ \boldsymbol{u}_2(k) \end{bmatrix}$$

$$= \begin{bmatrix} \boldsymbol{C}_{1a} & \boldsymbol{\epsilon} & \boldsymbol{D}_{1a,1b} & \boldsymbol{\epsilon} & \boldsymbol{D}_{1a,1a} & \boldsymbol{D}_{1a,2} \\ \boldsymbol{\epsilon} & \boldsymbol{I} & \boldsymbol{\epsilon} & \boldsymbol{\epsilon} & \boldsymbol{\epsilon} & \boldsymbol{\epsilon} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{\chi}_{a1}(k) \\ \boldsymbol{\chi}_{\lfloor \frac{i}{m} \rfloor a}(k) \\ \boldsymbol{\chi}_{b1}(k) \\ \boldsymbol{\chi}_{\lceil \frac{i}{m} \rceil b}(k) \\ \boldsymbol{u}_2(k) \end{bmatrix}$$

$$= \begin{bmatrix} \boldsymbol{C}_{1a} & [\boldsymbol{\epsilon}\ \boldsymbol{D}_{1a,1b}] & [\boldsymbol{\epsilon}\ \boldsymbol{D}_{1a,1a}] & \boldsymbol{D}_{1a,2} \\ \boldsymbol{\epsilon} & [\boldsymbol{I}\ \boldsymbol{\epsilon}] & \boldsymbol{\epsilon} & \boldsymbol{\epsilon} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{\chi}_a(k) \\ \boldsymbol{\chi}_b(k) \\ \boldsymbol{u}_2(k) \end{bmatrix}, \tag{18}$$

where $\boldsymbol{\epsilon} = -\infty$; similarly, substituting $\boldsymbol{y}_{1b}(k)$ from (14) yields:

$$\boldsymbol{\chi}_b(k+1) = \begin{bmatrix} \boldsymbol{C}_{1b} & [\boldsymbol{\epsilon}\ \boldsymbol{D}_{1b,1b}] & [\boldsymbol{\epsilon}\ \boldsymbol{D}_{1b,1a}] & \boldsymbol{D}_{1b,2} \\ \boldsymbol{\epsilon} & \boldsymbol{\epsilon} & [\boldsymbol{I}\ \boldsymbol{\epsilon}] & \boldsymbol{\epsilon} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{\chi}_a(k) \\ \boldsymbol{\chi}_b(k) \\ \boldsymbol{u}_2(k) \end{bmatrix}. \tag{19}$$

Adding $\boldsymbol{\chi}_a(k+1)$ from (18) and $\boldsymbol{\chi}_b(k+1)$ from (19) to the left-hand side of (14) and eliminating $\boldsymbol{y}_{1a}(k)$, $\boldsymbol{y}_{1b}(k)$, $\boldsymbol{u}_{1a}(k)$ and $\boldsymbol{u}_{1b}(k)$, which are captured by $\boldsymbol{\chi}_a(k+1)$, $\boldsymbol{\chi}_b(k+1)$, $\boldsymbol{\chi}_b(k)$ and $\boldsymbol{\chi}_a(k)$, respectively, from (14) yields the following results describing the canonical model of S after adding connection (y_1, u_1).

$$\begin{bmatrix} \boldsymbol{x}^{\mathcal{IO}}(k+1) \\ \boldsymbol{y}^{\mathcal{IO}}(k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}^{\mathcal{IO}} & \boldsymbol{B}^{\mathcal{IO}} \\ \boldsymbol{C}^{\mathcal{IO}} & \boldsymbol{D}^{\mathcal{IO}} \end{bmatrix} \otimes \begin{bmatrix} \boldsymbol{x}^{\mathcal{IO}}(k) \\ \boldsymbol{u}^{\mathcal{IO}}(k) \end{bmatrix}, \tag{20}$$

where

$$\boldsymbol{x}^{\mathcal{IO}}(k) = [\boldsymbol{x}(k)\ \boldsymbol{\chi}_a(k)\ \boldsymbol{\chi}_b(k)]^T,$$
$$\boldsymbol{u}^{\mathcal{IO}}(k) = \boldsymbol{u}_2(k),\ \boldsymbol{y}^{\mathcal{IO}}(k) = \boldsymbol{y}_2(k),$$
$$\boldsymbol{A}^{\mathcal{IO}} = \begin{bmatrix} \boldsymbol{A} & [\boldsymbol{\epsilon}\ \boldsymbol{B}_{1b}] & [\boldsymbol{\epsilon}\ \boldsymbol{B}_{1a}] \\ \begin{bmatrix} \boldsymbol{C}_{1a} \\ \boldsymbol{\epsilon} \end{bmatrix} & \begin{bmatrix} \boldsymbol{\epsilon} & \boldsymbol{D}_{1a,1b} \\ \boldsymbol{I} & \boldsymbol{\epsilon} \end{bmatrix} & \begin{bmatrix} \boldsymbol{\epsilon} & \boldsymbol{D}_{1a,1a} \\ \boldsymbol{\epsilon} & \boldsymbol{\epsilon} \end{bmatrix} \\ \begin{bmatrix} \boldsymbol{C}_{1b} \\ \boldsymbol{\epsilon} \end{bmatrix} & \begin{bmatrix} \boldsymbol{\epsilon} & \boldsymbol{D}_{1b,1b} \\ \boldsymbol{\epsilon} & \boldsymbol{\epsilon} \end{bmatrix} & \begin{bmatrix} \boldsymbol{\epsilon} & \boldsymbol{D}_{1b,1a} \\ \boldsymbol{I} & \boldsymbol{\epsilon} \end{bmatrix} \end{bmatrix}, \boldsymbol{B}^{\mathcal{IO}} = \begin{bmatrix} \boldsymbol{B}_2 \\ \begin{bmatrix} \boldsymbol{D}_{1a,2} \\ \boldsymbol{\epsilon} \end{bmatrix} \\ \begin{bmatrix} \boldsymbol{D}_{1b,2} \\ \boldsymbol{\epsilon} \end{bmatrix} \end{bmatrix},$$
$$\boldsymbol{C}^{\mathcal{IO}} = \left[\boldsymbol{C}_2\ [\boldsymbol{\epsilon}\ \boldsymbol{D}_{2,1b}]\ [\boldsymbol{\epsilon}\ \boldsymbol{D}_{2,1a}]\right] \text{ and } \boldsymbol{D}^{\mathcal{IO}} = \boldsymbol{D}_{2,2}.$$

In the special case when $mod(i,m) = 0$, all produced samples in an iteration are consumed in one iteration. This results in empty vectors $\boldsymbol{y}_{1b}(k)$ and $\boldsymbol{u}_{1a}(k)$ and $\boldsymbol{y}_{1a}(k) = \boldsymbol{y}_1(k)$ and $\boldsymbol{u}_{1a}(k) = \boldsymbol{u}_1(k)$; therefore, (13) and (20) become simpler.

Definition 3. [IO composite model] $CS = \mathcal{IO}(S, oi, m, i)$ is an operation taking a canonical model of system S, a connection $oi = (y_1, u_1)$ with $Sr(y_1) = Sr(u_1) = m$ and i representing the number of initial samples on oi, and returning the canonical model CS which is a composite model after adding connection oi to S. If $0 \le i < m$, the CS model can be calculated from (13); otherwise, from (20). In the former equation, the condition $\boldsymbol{D}_{1a,1b} = -\boldsymbol{\infty}_{(m-i)\times(m-i)}$ is sufficient to ensure deadlock-freedom.

Theorem 1. [Canonical model of a composite system] Given two M^2PLSs S and S' in canonical form with b connections between them. To determine the canonical-form representation of a composite system fabricated from those, first the consistency of the composite system is checked and its repetition vector $\boldsymbol{r}$ is calculated (see *Definition* 1). Then, operation $\mathcal{RS}$ (*Definition* 2) is performed to synchronize the sample rates on connections between those M^2PLSs. Finally, operation $\mathcal{IO}$ (*Definition* 3) is performed b times to find the canonical model after adding the b connections. Each step takes the calculated model of the pervious step and information of a new connection.

It is worth returning to the running example. After synchronizing and aggregating two systems, the canonical synchronized-rate model of $\mathcal{AS}$ was computed and shown in Fig. 4. The next step, as shown in Fig. 3 (b), is adding the connection from y_1 to u'_1 in Fig. 2 with $Is(y_1, u'_1) = 0$. Thus, the canonical model of the composite system after adding this connection can be calculated from (13). This is a special case and $\boldsymbol{y}_{1a}(k) = \boldsymbol{y}_1(k)$ and $\boldsymbol{u'}_{1a}(k) = \boldsymbol{u}'_1(k)$. The condition $\boldsymbol{D}_{1a,1b} = -\boldsymbol{\infty}_{2\times 2}$ is satisfied. Thus,

$$(\boldsymbol{A}^{\mathcal{RS}})^{\mathcal{IO}} = \boldsymbol{A}^{\mathcal{RS}} \oplus \boldsymbol{B}_1^{\mathcal{RS}} \otimes \boldsymbol{C}_1^{\mathcal{RS}} = \begin{bmatrix} 2 & \epsilon & \epsilon & \epsilon \\ 4 & 2 & \epsilon & \epsilon \\ 11 & 12 & 8 & \epsilon \\ 12 & 13 & 9 & 2 \end{bmatrix},$$

$$(\boldsymbol{B}^{\mathcal{RS}})^{\mathcal{IO}} = \boldsymbol{B}_2^{\mathcal{RS}} \oplus \boldsymbol{B}_1^{\mathcal{RS}} \otimes \boldsymbol{D}_{1,2}^{\mathcal{RS}} = \begin{bmatrix} \epsilon & \epsilon & 2 & \epsilon & \epsilon \\ 2 & 1 & 4 & \epsilon & \epsilon \\ 9 & 5 & 11 & 6 & 2 \\ 10 & 6 & 12 & 7 & 3 \end{bmatrix},$$

$$(\boldsymbol{C}^{\mathcal{RS}})^{\mathcal{IO}} = \boldsymbol{C}_2^{\mathcal{RS}} \oplus \boldsymbol{D}_{2,1}^{\mathcal{RS}} \otimes \boldsymbol{C}_1^{\mathcal{RS}} = \begin{bmatrix} 2 & \epsilon & \epsilon & \epsilon \\ 8 & 6 & 5 & 1 \\ 5 & 3 & 2 & \epsilon \\ 12 & 10 & 9 & 2 \\ 9 & 7 & 6 & \epsilon \end{bmatrix},$$

$$\text{and } (\boldsymbol{D}^{\mathcal{RS}})^{\mathcal{IO}} = \boldsymbol{D}_{2,2}^{\mathcal{RS}} \oplus \boldsymbol{D}_{2,1}^{\mathcal{RS}} \otimes \boldsymbol{D}_{1,2}^{\mathcal{RS}} = \begin{bmatrix} \epsilon & \epsilon & 2 & \epsilon & \epsilon \\ 6 & \epsilon & 8 & 3 & \epsilon \\ 3 & \epsilon & 5 & \epsilon & \epsilon \\ 10 & 6 & 12 & 7 & 3 \\ 7 & 3 & 9 & 4 & \epsilon \end{bmatrix}.$$

These matrices describe the canonical model of CS in Fig. 2. This model in turn is used to compute the canonical model of CS' as it is conveyed in Fig. 3 (c). According to *Definition* 3, $CS' = \mathcal{IO}(CS, (y'_1, u_1), 2, 2)$ which is determined from (20). As a result of $mod(i,m) = 0$, $\boldsymbol{y'}_{1a}(k) = \boldsymbol{y'}_1(k)$ and $\boldsymbol{u}_{1a}(k) = \boldsymbol{u}_1(k)$. Since $\lfloor \frac{i}{m} \rfloor = 1$, then $\boldsymbol{\chi}_a(k+1) = \boldsymbol{\chi}_{1a}(k+1) = \boldsymbol{y'}_1(k)$ and $\boldsymbol{\chi}_a(k+1) = \boldsymbol{u}_1(k)$.

For this special case,

$$((\boldsymbol{A}^{\mathcal{RS}})^{\mathcal{IO}})^{\mathcal{IO}} = \begin{bmatrix} (\boldsymbol{A}^{\mathcal{RS}})^{\mathcal{IO}} & (\boldsymbol{B}^{\mathcal{RS}})^{\mathcal{IO}}_1 \\ (\boldsymbol{C}^{\mathcal{RS}})^{\mathcal{IO}}_1 & (\boldsymbol{D}^{\mathcal{RS}})^{\mathcal{IO}}_{1,1} \end{bmatrix} = \begin{bmatrix} 2 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ 4 & 2 & \epsilon & \epsilon & 2 & 1 \\ 11 & 12 & 8 & \epsilon & 9 & 5 \\ 12 & 13 & 9 & 2 & 10 & 6 \\ 8 & 6 & 5 & 1 & 6 & \epsilon \\ 12 & 10 & 9 & 2 & 10 & 6 \end{bmatrix},$$

where, as an example, $(\boldsymbol{C}^{\mathcal{RS}})^{\mathcal{IO}}_1$ is a matrix constructed from the second and third rows of $(\boldsymbol{C}^{\mathcal{RS}})^{\mathcal{IO}}$ indicating the interplay between y'_1 and x.

$$((\boldsymbol{B}^{\mathcal{RS}})^{\mathcal{IO}})^{\mathcal{IO}} = \begin{bmatrix} (\boldsymbol{B}^{\mathcal{RS}})^{\mathcal{IO}}_2 \\ (\boldsymbol{D}^{\mathcal{RS}})^{\mathcal{IO}}_{1,2} \end{bmatrix} = \begin{bmatrix} 2 & 4 & 11 & 12 & 5 & 9 \\ \epsilon & \epsilon & 6 & 7 & \epsilon & 4 \\ \epsilon & \epsilon & 2 & 3 & \epsilon & \epsilon \end{bmatrix}^T,$$

$$((\boldsymbol{C}^{\mathcal{RS}})^{\mathcal{IO}})^{\mathcal{IO}} = \begin{bmatrix} (\boldsymbol{C}^{\mathcal{RS}})^{\mathcal{IO}}_2 & (\boldsymbol{D}^{\mathcal{RS}})^{\mathcal{IO}}_{2,1} \end{bmatrix} = \begin{bmatrix} 2 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ 5 & 3 & 2 & \epsilon & 3 & \epsilon \\ 9 & 7 & 6 & \epsilon & 7 & 3 \end{bmatrix}$$

$$\text{and } ((\boldsymbol{D}^{\mathcal{RS}})^{\mathcal{IO}})^{\mathcal{IO}} = (\boldsymbol{D}^{\mathcal{RS}})^{\mathcal{IO}}_{2,2} = \begin{bmatrix} 2 & \epsilon & \epsilon \\ 5 & \epsilon & \epsilon \\ 9 & 4 & \epsilon \end{bmatrix}.$$

In contrast to the shown model in Fig. 2, the above calculated model, which is in the canonical form, can be utilized to control and analyze the system. For instance, the matrices are used to evaluate the throughput and latency of the system by adopting the method of Geilen et al. (2020).

6. CONCLUSION

In this paper, an algebraic compositional model of M^2PLSs in canonical form was introduced. The proposed method can find the canonical model of any system constructed from canonical-form representations of M^2PLSs. A check for consistency of the composite system, which is a necessary condition for a system to be modeled, and two operations were explained, (I) rate synchronization and (II) IO composition. The first operation synchronizes the rates of two M^2PLSs on connections between them and aggregates the two systems into one system, while the latter operation computes the canonical model of a composite system after adding an IO connection. Having a deadlock-free composite system is a necessary condition for the second operation. A sufficient condition for deadlock-freeness is given.

One use of the proposed method is that it facilitates (re-)calculating canonical models of reconfigurable as well as of composite systems. Consider a system with multiple configurations, of which the model dynamically changes. To evaluate its performance properties, or to control the system, its canonical model should be (re-)determined for every configuration. Changes from one configuration to another might only appear in a part of the system. Saving M^2PLSs in a repository and (re-)computing the compositional model of the system based on its configuration, instead of determining the canonical model from scratch, may considerably reduce the cost of modeling.

As future work, we aim to adopt the proposed method to design and analyze video processing pipelines. This method will be used to reason about performance properties of complex streaming applications with multiple configurations. This will simplify the quality and resource management of these applications.

REFERENCES

Aguado, J., Mendler, M., Wang, J.J., Bodin, B., and Roop, P. (2017). Compositional timing-aware semantics for synchronous programming. In *IEEE Forum on Specification and Design Languages (FDL)*, 1–8.

Alizadeh Ara, H., Geilen, M., Behrouzian, A., Basten, T., and Goswami, D. (2018). Compositional dataflow modelling for cyclo-static applications. In *IEEE 21st Euromicro Conference on Digital System Design (DSD)*, 121–129.

Baccelli, F., Cohen, G., Olsder, G., and Quadrat, J. (1992). *Synchronization and linearity: an algebra for discrete event systems.* Wiley, Chichester, New York.

Cruz, R. (1991). A calculus for network delay. II. Network analysis. *IEEE Transactions on Information Theory*, 37(1), 132–141.

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

FitOptiVis (2019). The FitOptiVis ECSEL project: highly efficient distributed embedded image/video processing in cyberphysical systems. In *ACM Int'l Conf. on Computing Frontiers (CF)*, 333–338. ACM.

Geilen, M.C.W., Skelin, M., van Kampenhout, J.R., Alizadeh Ara, H., Basten, T., Stuijk, S., and Goossens, K.G.W. (2020). Scenarios in dataflow modeling and analysis. In *System-Scenario-based Design Principles and Applications.* Springer.

Ghamarian, A., Geilen, M., Stuijk, S., Basten, T., Theelen, B., Mousavi, M., Moonen, A., and Bekooij, M. (2006). Throughput analysis of synchronous data flow graphs. In *6th International Conference on Application of Concurrency to System Design (ACSD)*, 25–36. IEEE.

Heidergott, B., Olsder, G.J., and Van der Woude, J. (2005). *Max Plus at Work: Modeling and Analysis of Synchronized Systems: A Course on Max-Plus Algebra and Its Applications.* Princeton University Press.

Imaev, A. and Judd, R.P. (2008). Hierarchial modeling of manufacturing systems using max-plus algebra. In *American Control Conference*, 471–476.

Kersbergen, B., Rudan, J., Van den Boom, T., and De Schutter, B. (2016). Towards railway traffic management using switching max-plus-linear systems. *Discrete Event Dynamic Systems*, 26(2), 183–223.

Lee, E. (1991). Consistency in dataflow graphs. In *International Conference on Application Specific Array Processors (ASAP)*, 355–369. IEEE.

Seleim, A. and ElMaraghy, H. (2014). Max-plus modeling of manufacturing flow lines. *Procedia CIRP series in Variety Management in Manufacturing*, 17, 71–75.

Skelin, M. and Geilen, M.C.W. (2017). Towards component-based (max,+) algebraic throughput analysis of hierarchical synchronous data flow models. In *Proc. SAFECOMP 2017 Workshops*, 462–476. Springer.

Stuijk, S., Basten, T., Geilen, M., and Corporaal, H. (2007). Multiprocessor resource allocation for throughput-constrained synchronous dataflow graphs. In *44th ACM/IEEE Design Automation Conference (DAC)*, 777–782.