2010 American Control Conference
Marriott Waterfront, Baltimore, MD, USA
June 30-July 02, 2010

Signal Flow Graphs over Max-Plus Algebra and Applications

Aleksey Imaev and Robert P. Judd

Abstract— **This work investigates properties of signal flow graphs (SFGs) over max-plus algebra, which are referred to as synchronous SFGs. New topological methods for evaluating gains of synchronous matrix SFGs are described. These methods are based on the existing theory for matrix SFGs over regular algebra. SFGs are useful in studying complex engineering systems by representing them as interconnection of relatively simple subsystems. In particular, synchronous SFGs can be used to graphically model the timing behavior of deterministic discrete event systems. The paper illustrates an application of synchronous SFGs to modeling and performance evaluation of deterministic manufacturing systems.**

I. Introduction

A (scalar) *signal flow graph* (SFG), originally developed by Mason [1], is a graphical representation of a set of linear equations. In contrast, a *matrix* signal flow graph (MSFG) is a pictorial representation of a set of linear *matrix* equations.

This paper focuses on SFGs over max-plus algebra. Max-plus algebra is an algebra with only two operations, namely maximization, denoted by $\oplus$, and addition, denoted by $\otimes$, which are defined for the elements in $\mathbb{R}_{\max} = \{\mathbb{R} \cup -\infty\}$, where $\mathbb{R}$ is the set of all real numbers. For elements $a, b \in \mathbb{R}_{\max}$, we have $a \oplus b = \max(a, b)$ and $a \otimes b = a + b$. Operation $\oplus$ has null element, $\varepsilon = -\infty$, since $a \oplus \varepsilon = a$. Similarly operation $\otimes$ has unit element, $e = 0$, as $a \otimes e = a$. Throughout the paper, a SFG over max-plus algebra is referred to as a *synchronous SFG*, because maximization operation represents synchronization phenomena in discrete event systems. In contrast, a SFGs over regular algebra is referred to as a *regular* SFG.

Pliam [2] have shown that the classic gain formula of Mason [1] is valid only for SFGs over commutative rings (e.g., regular SFGs). Gains of signal flow graphs over non-commutative ring (e.g. regular MSFGs) can be graphically evaluated using either a Riegle's formula [3] or topological procedures described in [4]. A theory of synchronous MSFGs has not been addressed yet in the literature. This paper presents new topological methods for evaluating gains of synchronous MSFGs.

A *Discrete Event System* (DES) is a system, which is characterized by a set of states and a set of events [5]. Events cause the DES to change its state at discrete time instants. A DES is called *deterministic* if its future states are uniquely determined by the current state and the external inputs [6, Ch.1]. Max-plus algebra is an attractive tool for modeling of deterministic DESs because the event timing dynamics of these systems can be expressed by a set of equations in max-plus algebra. The max-plus algebraic model of a deterministic DES is usually obtained from the timed event graph model of the system [7], [8].

In [9], [10] we proposed a modeling approach for deterministic manufacturing systems, which is based on block diagrams. A block can be a single manufacturing operation, a single machine, a single part or a factory. Each block has three inputs and three outputs and is represented by a set of linear max-plus algebraic equations. A complex manufacturing system can be modeled as a composition of simpler manufacturing blocks. The model is hierarchial – a network of blocks can be combined into one block that has the same input/output structure. The model is graphically represented by a synchronous MSFG. In [9] the model was analyzed using algebraic methods, in this paper we focus on topological methods based on synchronous MSFGs.

Section II provides background information on max-plus algebra. Section III introduces definitions for synchronous MSFGs. Section IV illustrates an essential difference between regular and synchronous MSFGs. Topological methods for evaluating gains of synchronous MSFGs are presented in Section V. Section VI illustrates an application of synchronous MSFGs to modeling of deterministic manufacturing systems. Conclusions are presented in Section VII.

II. Max-plus Algebra Basics

A comprehensive review of max-plus algebra can be found in [11], [8].

Max plus algebra is extended to matrices in the same way as conventional algebra but with + replaced by $\oplus$ and $\times$ replaced by $\otimes$. A set of all $n \times m$ matrices is denoted by $\mathbb{R}_{\max}^{m \times n}$. We say that an $n \times m$ matrix $\mathbf{A}$ exists if and only if $\mathbf{A} \in \mathbb{R}_{\max}^{n \times m}$. Note that, similar to regular algebra, matrix multiplication in max-plus algebra is noncommutative.

Analogous to conventional algebra $\otimes$ is assumed precedence over $\oplus$ and if it is clear that the $\otimes$ symbol is used it is sometimes omitted, *i.e.* $\mathbf{A} \oplus \mathbf{BC}$ should be understood as $\mathbf{A} \oplus (\mathbf{B} \otimes \mathbf{C})$. Throughout the paper all the equations are assumed to be written in terms of max-plus algebra unless stated otherwise.

For any square matrix $\mathbf{A} \in \mathbb{R}_{\max}^{n \times n}$, $\mathbf{A}$ in n^{th} power is defined by $\mathbf{A}^n = \mathbf{A} \otimes \mathbf{A} \otimes \ldots \mathbf{A} \to n$ times. Define Kleene star operator on $\mathbf{A}$ denoted by $\mathbf{A}^\star$ as

$$\mathbf{A}^\star = \mathbf{A}^e \oplus \mathbf{A}^1 \oplus \ldots \oplus \mathbf{A}^\infty = \bigoplus_{k=0}^{\infty} \mathbf{A}^k,$$

A. Imaev, is with the School of Electrical Engineering and Computer Science, Ohio University, Athens, Ohio 45701, USA ai400695@ohio.edu

R.P. Judd is a Professor and Chair of the Department of Industrial and Systems Engineering, Ohio University, Athens, OH 45701, USA judd@ohio.edu

where $\mathbf{A}^e = \mathbf{E}$ and $\mathbf{E} \in \mathbb{R}_{\max}^{n\times n}$ refers to identity matrix which has e's on the main diagonal and ε's elsewhere.

Theorem 2.1: [11, Theorem 2.10] $\mathbf{x} = \mathbf{A}^\star \otimes \mathbf{b}$ is the minimum solution to the equation $\mathbf{x} = \mathbf{A} \otimes \mathbf{x} \oplus \mathbf{b}$, provided that $\mathbf{A}^\star$ exists.

III. Synchronous MSFGs - Preliminary Definitions

A directed graph G is defined as an ordered pair (N, E) where N is a finite set of nodes and E is a set of ordered pairs of nodes called arcs. A pair (i, j) denotes an arc directed from node i to node j. An *arc progression* p from node x_1 to x_k in a directed graph is a sequence of arcs that connects a sequence of nodes, i.e. $p = \{(x_1, x_2)(x_2, x_3)\dots(x_{k-1}, x_k)\}$ is an arc progression. A *path* is an arc progression in which no node appears more than once. A path in which the terminal node and the initial node are the same node is a loop. An arc from a node to itself is called a *self-loop*.

A synchronous MSFG is a directed graph in which every node is associated with a vector $\mathbf{x}_j$ and every arc is associated with a matrix $\mathbf{A}_{i,j}$ such that for every node $\mathbf{x}_j$, there corresponds the matrix equation in max-plus algebra $\mathbf{x}_j = \bigoplus(\mathbf{A}_{j,i}\mathbf{x}_i)$, where the summation is over all arcs terminating on $\mathbf{x}_j$. A synchronous MSFG topologically portrays a set of linear matrix equations in max-plus algebra.

Given a MSFG, a node that has no terminating arcs is referred to as a source node, and a node that has no originating arcs is referred to as a sink node. The rectangular matrix associated with an arc is called the *transmittance* of the arc. Throughout the paper, it is assumed that if in a arc in a graph is not labeled with the weight, then its transmittance equals $\mathbf{E}$ (an identity matrix in max-plus algebra of appropriate dimensions).

The *graph gain* or *graph transmission* from the source node $\mathbf{x}_i$ to the sink node $\mathbf{x}_j$ is defined by the matrix $\mathbf{T}_{j,i}$, which relates $\mathbf{x}_i$ to $\mathbf{x}_j$ by $\mathbf{x}_j = \mathbf{T}_{j,i}\mathbf{x}_i$.

A note on the notation. For a positive integer K, define $\underline{K} = \{1, 2, \dots, K\}$. Let $\mathbf{s}$ be an ordered set or a vector. Then $|\mathbf{s}|$ gives the number of elements in $\mathbf{s}$. The i-th element of $\mathbf{s}$ is denoted by $[\mathbf{s}]_i$, for any $i \in \underline{|\mathbf{s}|}$. Similarly, given a matrix $\mathbf{A}$, $[\mathbf{A}]_{j,i}$ refers to the element of $\mathbf{A}$ in j-th row and i-th column. In contrast, $\mathbf{T}_{j,i}$ is a matrix.

IV. Regular vs. Synchronous MSFGs

Regular MSFGs belong to the class of SFGs over non-commutative ring [12]. The classic gain formula of Mason applies only to SFGs over commutative rings [2]. Gains of MSFGs can be evaluated using a formula developed by Riegle [3]. In addition, there are three other topological methods for evaluating gains of regular MSFGs [3]: a) by repeated application of basic graph reduction rules, b) by the return loop method, and c) by optimal topological procedure.

The theory of synchronous MSFGs has not been studied before. The gain methods for regular MSFGs cannot be applied to synchronous MSFGs directly. This is because topological methods for evaluating gains of a regular MSFGs require additive inverse operations as well as evaluating

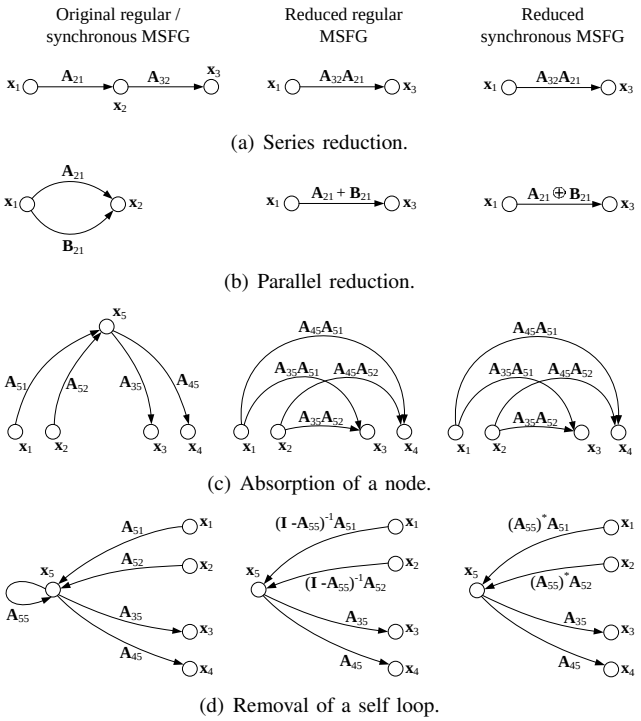

Fig. 1. An illustration of the basic reduction rules.

matrix inverses, when the graph contains loops. These operations are not directly defined in max-plus algebra. [1] Hence, the difference between regular and synchronous MSFGs lies in the treatment of loops as described below. Consider the following equation in regular algebra

$$\mathbf{x}_2 = \mathbf{A}\mathbf{x}_2 + \mathbf{B}\mathbf{x}_1. \tag{1}$$

Solving for $\mathbf{x}_2$ we get $\mathbf{x}_2 = (\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{x}_1$, where $\mathbf{I}$ is the identity matrix. Now, consider the analog of (1) in max-plus (dioid) algebra:

$$\mathbf{x}_2 = \mathbf{A}\mathbf{x}_2 \oplus \mathbf{B}\mathbf{x}_1. \tag{2}$$

From Theorem 2.1, its minimum solution is $\mathbf{x}_2 = \mathbf{A}^\star\mathbf{B}\mathbf{x}_1$. Therefore, $\mathbf{A}^\star$ in max-plus algebra is analogous to $(\mathbf{I}-\mathbf{A})^{-1}$ in regular algebra.

V. Topological Methods for Evaluating Gains of Synchronous MSFGs

Using the fact that $\mathbf{A}^\star$ in max-plus algebra is analogous to $(\mathbf{I} - \mathbf{A})^{-1}$ in regular algebra, the gain methods for regular MSFGs can be straightforwardly extended to synchronous MSFGs. The following develops extensions of (a) basic block diagram reduction rules and (b) the return loop method to synchronous MSFGs.

A. Basic Graph Reduction Rules

By the successive application of basic graph reduction rules, a synchronous MSFG can be reduced to a simplified graph from which the desired graph gains can be obtained directly. The basic graph reduction rules for both synchronous

[1] The residuation theory in max-plus algebra can deal with inverse problems [7]

and regular MSFGs are illustrated in Figure 1. It should be noted that the expressions for the transmittances of arcs of regular (synchronous) MSFGs in Figure 1 are written in regular (max-plus) algebra. The following lists the basic graph reduction rules for synchronous MSFGs.

1) Series reduction is illustrated in Figure 1(a).
2) Parallel reduction is illustrated in Figure 1(b).
3) Absorption of a node is illustrated in Figure 1(c). The node $\mathbf{x}_5$ is absorbed.
4) Removal of a self-loop is illustrated in Figure 1(d). The self-loop $\mathbf{A}_{55}$ at node $\mathbf{x}_5$ is removed. Note the difference between removal of a self-loop in regular vs. synchronous MSFG.

Proofs are straightforward and omitted (e.g. refer to [3] for the case of regular MSFGs and refer to the previous section for the removal of a self-loop in synchronous MSFGs).

B. Return Loop Method

The return loop method is an alternative to the repeated application of the basic reduction rules. The original return loop method for regular MSFGs is described in [4]. In this section, an extension of the return loop method to synchronous MSFGs is described. First we introduce some preliminary definitions. Let p be a path from an input node k to an output node j.

- The *path product* of p is the product of arc transmittances of p multiplied in reverse order from node j to node k.
- A node in a MSFG is said to be *split* when it is replaced by two nodes, a source node and a sink node such that arcs terminating on the original node are made to terminate on the new sink and all the arcs outgoing from the original node are made to originate at the new source [4].
- The *node transmission* $\mathbf{N}_i$ of a node i is the graph transmission between the source and the sink which is created by splitting the node i.
- The *node transmission* $\mathbf{N}_i^p$ of a node i on path p (from k to j) is $\mathbf{N}_i$ calculated under the condition that all nodes on p between node i and the output node j are split.
- The *node factor* $\hat{\mathbf{N}}_i^p$ of a node i on path p is $(\mathbf{N}_i^p)^\star$.

The following is the *return loop method* that yields the graph transmission (graph gain) between the source node k and the sink node j.

1) Find all of the paths from the source node k to the sink node j.
2) The contribution of the gain by a path p is equal to the path product of p interrupted by the node factor of every node on the path p. The node factor of node i is inserted between the arc transmittances of the path product touching node i.
3) The graph gain $T_{j,k}$ is equal to the (max-plus algebraic) sum of the contributions of the gain by each path from i to j, where the summation is over all such paths.

The proof of the return loop method for synchronous MSFGs depends entirely on the proof presented by Riegle and Lin [4] for regular MSFGs. The only noteworthy dissimilarity between the methods is as follows. For regular MSFGs the node factor of the node i on path p is defined as $(\mathbf{I}-\mathbf{N}_i^p)^{-1}$ (where $\mathbf{I}$ is the identity matrix in regular algebra), whereas for synchronous MSFGs the node factor of the node i on p is defined as $(\mathbf{N}_i^p)^\star$.

VI. Application of Synchronous MSFGs to Modeling of Manufacturing Systems

A deterministic manufacturing systems can be modeled by a synchronous MSFG using the hierarchical approach proposed in [9]. In [9] the approach was analyzed using algebraic methods, in this paper we focus on topological methods based on synchronous MSFGs.

A. Manufacturing Block

A manufacturing system is modeled by a generic block with three inputs and three outputs. In order to operate, the system requires a set of parts and a set of resources. After the system has processed the parts and the resources, they are are released by the system. Let $\mathbf{m}$ denote an ordered set of system's resources, such as machines, buffers, etc. Let $\mathbf{n}^{in}$ be the ordered set of parts that enter the system and let $\mathbf{n}^{out}$ be the ordered set of parts that leave the system. The order of elements in either $\mathbf{m}$, $\mathbf{n}^{in}$ or $\mathbf{n}^{out}$ can be chosen arbitrary. Let $i \in \{1,2,\ldots,|\mathbf{n}^{in}|\}$, $k \in \{1,2,\ldots,|\mathbf{m}|\}$ and $j \in \{1,2,\ldots,|\mathbf{n}^{out}|\}$. The inputs and outputs of the block are vector valued. The inputs, $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{w}$ are defined as

- $[\mathbf{u}]_i$ is the time when $[\mathbf{n}^{in}]_i$ becomes available for the system;
- $[\mathbf{v}]_j$ is the time when $[\mathbf{n}^{out}]_j$ is removed from the system;
- $[\mathbf{w}]_k$ is the time when $[\mathbf{m}]_k$ becomes available for the system.

The outputs, $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ are defined as:

- $[\mathbf{x}]_j$ is the time when $[\mathbf{n}^{out}]_j$ is ready to leave the system;
- $[\mathbf{y}]_i$ is the time when $[\mathbf{n}^{in}]_i$ actually enters the system;
- $[\mathbf{z}]_k$ is the time when $[\mathbf{m}]_k$ is "set free" by the system.

Since the system is deterministic, the relationship between the input and the output can be expressed by the following equation in max-plus algebra:

$$\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\\ \mathbf{z}\end{bmatrix} = \begin{bmatrix}\mathbf{F}_{xu} & \mathbf{F}_{xv} & \mathbf{F}_{xw}\\ \mathbf{F}_{yu} & \mathbf{F}_{yv} & \mathbf{F}_{yw}\\ \mathbf{F}_{zu} & \mathbf{F}_{zv} & \mathbf{F}_{zw}\end{bmatrix} \begin{bmatrix}\mathbf{u}\\ \mathbf{v}\\ \mathbf{w}\end{bmatrix} = \mathbf{F} \begin{bmatrix}\mathbf{u}\\ \mathbf{v}\\ \mathbf{w}\end{bmatrix}, \qquad (3)$$

where $\mathbf{F}$ is a matrix that describes input-output relation. It is clear that (3) can be graphically represented by a synchronous MSFG with three inputs and three outputs.

B. Interconnection of Blocks

Let S_c be a system composed from a set of M manufacturing subsystems $\{S_1, S_2, \ldots, S_M\}$. Let $\mathbf{m}_c$, $\mathbf{n}_c^{in}$, $\mathbf{n}_c^{out}$ be ordered sets of resources and parts associated with S_c. Let

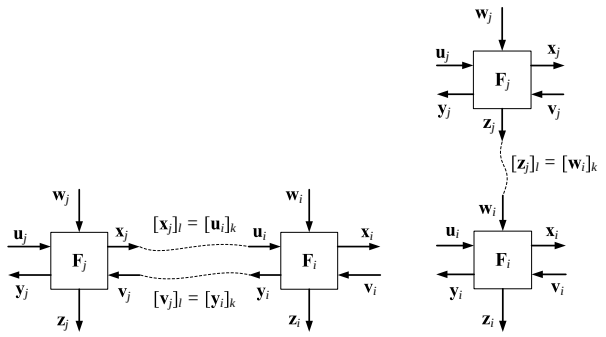

Fig. 2. Interconnection of blocks: (a) part-flow interconnection and (b) machine-flow interconnection.

the inputs and the outputs of S_c, namely $\mathbf{u}_c$, $\mathbf{v}_c$, $\mathbf{w}_c$ and $\mathbf{x}_c$, $\mathbf{y}_c$, $\mathbf{z}_c$, be defined with respect to $\mathbf{m}_c$, $\mathbf{n}_c^{in}$, $\mathbf{n}_c^{out}$.

The blocks are interconnected through part-flow and machine flow interconnections. Consider a part n, which enters S_i from an upstream S_j. Then we have $[\mathbf{v}_j]_l = [\mathbf{y}_i]_k$, as shown in Figure 2(a). This type of horizontal interconnection of blocks is referred to as *part-flow* interconnection. Likewise, consider a resource m, which is first used by S_j and then it is used by S_i. Then we have $[\mathbf{w}_i]_k = [\mathbf{z}_j]_l$ as shown in Figure 2(b). This type of vertical interconnection of blocks is referred to as *resource-flow* interconnection.

C. Basic Manufacturing Blocks

In this subsection timing models of basic manufacturing blocks are presented, namely the models of:(a) single resource manufacturing a part, and (b) unit capacity buffer storing a part. A complete discussion of basic manufacturing blocks is presented in [9].

1) Single machine processing single part: Consider machine m processing part n. Let t be processing time of n on m. Suppose that the system is modeled by using equation of the form (3) having inputs u, v, w and outputs x, y, z, which are all scalars because there is only one resource and one part. The part n enters the system as soon as both m and n are available, therefore $y = u \oplus w$. The part is ready to leave the system as soon as its processing is done on the machine, therefore $x = t(u \oplus w)$. The machine is "set free" by the system as soon as n is removed from the system, therefore $z = v$. Synchronous MSFG of the system is provided in Figure 3(a).

2) Unit capacity buffer: McCormick et al. [13] show that a buffer of unit capacity can be represented by a resource having zero processing time for jobs that enter the buffer. Therefore for buffer of unit capacity the synchronous MSFG in Figure 3(a) becomes the one shown in Figure 3(b).

D. Line Applications

Line applications include models of a single part processed by a set of M resources and models of a single resource processing a set of N parts. They are called line applications because in the former case basic blocks are stacked horizontally and in the latter case basic blocks are stacked vertically.

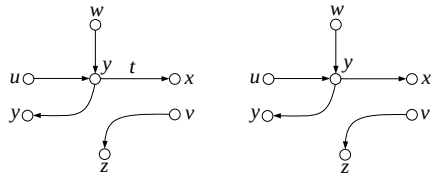

Fig. 3. Basic manufacturing blocks

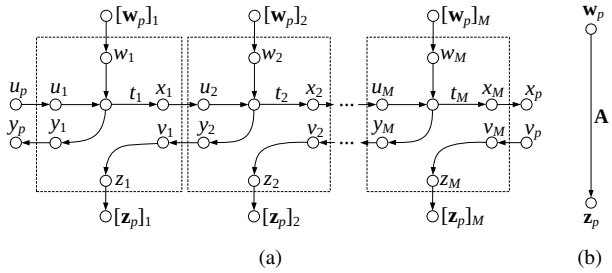

Fig. 4. Model of a part processed by a set of M machines. (a) portrays how individual operations can be stacked together horizontally, (b) shows a reduced version of the SMSFG assuming that $u_p = \varepsilon$.

1) One Part and M Resources: Consider a system consisting of a part processed by a set of M resources $\{m_1, m_2, \ldots, m_M\}$. It is assumed that there are no buffers between the machines. Let t_i be processing time of the part on machine m_i. An operation of the part being processed on m_i is modeled by a synchronous SFG of the form shown in Figure 3(a), with inputs u_i, v_i and w_i and outputs x_i, y_i and z_i. The system can be represented by a sequence of blocks sacked horizontally, where inputs and outputs are

$$u_p = u_1, \quad \mathbf{w}_p = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_M \end{bmatrix}, \quad \mathbf{x}_p = x_M, \quad \mathbf{z}_p = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_M \end{bmatrix}.$$

as shown in the synchronous SFG in Figure 4(a). We assume that the part is always available to the system, i.e. $\mathbf{u}_p = \varepsilon$.

The graph in Figure 4(a) can be reduced to the synchronous MSFG shown in Figure 4(b), where

$$\mathbf{A} = \begin{bmatrix} t_1 & e & \varepsilon & & \varepsilon \\ t_1t_2 & t_2 & e & & \varepsilon \\ t_1t_2t_3 & t_2t_3 & t_3 & & \varepsilon \\ \vdots & \vdots & \vdots & \ddots & \\ t_1t_2\ldots t_M & t_2t_3\ldots t_M & t_3t_4\ldots t_M & \ldots & t_M \end{bmatrix}.$$

Note that $\mathbf{A}$ is determined graphically from the graph in Figure 4(a). For example $[\mathbf{A}]_{2,1} = t_1t_2$ is the gain of the graph from $[\mathbf{w}_p]_1$ to $[\mathbf{z}_p]_2$.

2) N Parts and One Resource: Consider a system consisting of a machine that processes a set of N parts $\{n_1, n_2, \ldots, n_N\}$. It is assumed that there are no buffers in the system. Let t_i be processing time of n_i on the machine.

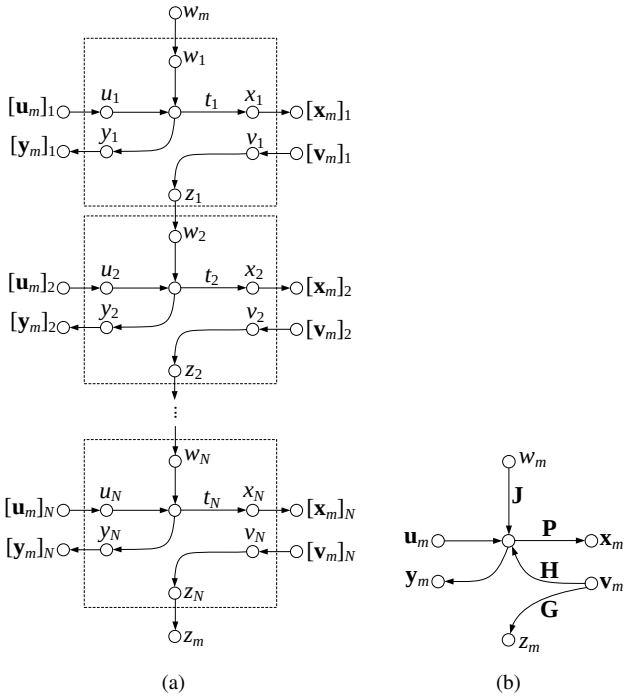

Fig. 5. A machine processing a set of parts: (a) portrays how individual operations can be stacked together vertically, (b) shows a more compact version of the synchronous MSFG.

Then operation of part n_i being processed on the machine is modeled by a basic synchronous SFG of the form shown in Figure 3(a), with inputs u_i, v_i and w_i and outputs x_i, y_i and z_i. The system can be represented by a sequence of blocks stacked vertically as shown in the synchronous SFG in Figure 5(a). Note that the inputs and outputs of the system are as follows

$$\mathbf{u}_m = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix}, \quad w_m = w_1, \quad \mathbf{x}_m = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \end{bmatrix}, \quad z_m = z_N.$$

The graph in Figure 5(a) can be reduced to the synchronous MSFG shown in Figure 5(b), where

$$\mathbf{P} = \begin{bmatrix} t_1 & \varepsilon & & \varepsilon \\ \varepsilon & t_2 & & \varepsilon \\ & & \ddots & \\ \varepsilon & \varepsilon & & t_M \end{bmatrix}, \mathbf{H} = \begin{bmatrix} \varepsilon & & \varepsilon & \varepsilon \\ e & & \varepsilon & \varepsilon \\ & \ddots & & \\ \varepsilon & & e & \varepsilon \end{bmatrix},$$

$$\mathbf{J} = \begin{bmatrix} e \\ \varepsilon \\ \vdots \\ \varepsilon \end{bmatrix}, \mathbf{G} = \begin{bmatrix} \varepsilon & \dots & \varepsilon & e \end{bmatrix}.$$

E. Modeling a Permutation Flow Shop

Consider a flow-shop system with M machines. The system is supposed to produce N parts. It is assumed that there are no buffers between the machines. Let $\mathbf{m} = [m_1, m_2, \dots, m_M]$ be an ordered set of machines and $\mathbf{n} = [n_1, n_2, \dots, n_N]$ be an ordered set of jobs. Each $n_i \in \mathbf{n}$ is processed by the machines in the order specified by $\mathbf{m}$. Each machine processes parts according to sequence $\mathbf{n}$ – this sequence is the same for all the machines in the system. Each job $n_j \in \mathbf{n}$ requires a processing time $t_{i,j}$ on machine $m_i \in \mathbf{m}$.

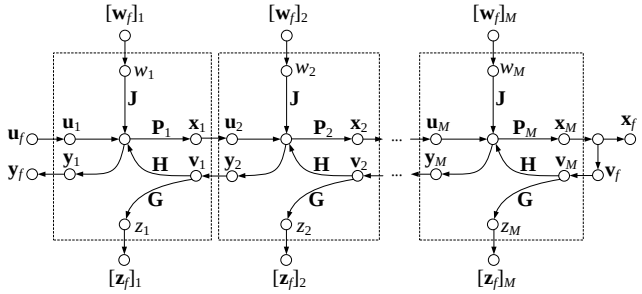

Fig. 6. Synchronous MSFG representation of a permutation flow shop. Each $\mathcal{G}_i$, $i \in \{1, 2, \dots, M\}$, models $m_i \in \mathbf{m}$ processing a sequence of parts $\mathbf{n}$.

A manufacturing process of machine $m_i \in \mathbf{m}$ processing a sequence of parts $\mathbf{n}$ can be modeled by a synchronous MSFG, $\mathcal{G}_i$, of the form shown in Figure 5(b), where

$$\mathbf{P}_i = \begin{bmatrix} t_{i,1} & \varepsilon & & \varepsilon \\ \varepsilon & t_{i,2} & & \varepsilon \\ & & \ddots & \\ \varepsilon & \varepsilon & & t_{i,M} \end{bmatrix}, \mathbf{H} = \begin{bmatrix} \varepsilon & & \varepsilon & \varepsilon \\ e & & \varepsilon & \varepsilon \\ & \ddots & & \\ \varepsilon & & e & \varepsilon \end{bmatrix}.$$

The flow shop system can be represented as a serial interconnection of synchronous MSFGs $\mathcal{G}_i$, $i \in \{1, 2, \dots, M\}$ as illustrated in Figure 6. It is assumed that the parts are removed from the system as soon as they are ready to leave it, therefore $\mathbf{v}_f = \mathbf{x}_f$.

Example Consider a permutation flow shop with 3 machines processing 4 parts. Processing time matrix for the system is

$$\mathbf{T} = \begin{bmatrix} t_{1,1} & t_{1,2} & t_{1,3} & t_{1,4} \\ t_{2,1} & t_{2,2} & t_{2,3} & t_{2,4} \\ t_{3,1} & t_{3,2} & t_{3,3} & t_{3,4} \end{bmatrix} = \begin{bmatrix} 3 & 2 & 1 & 2 \\ 2 & 3 & 1 & 1 \\ 1 & 4 & 2 & 1 \end{bmatrix}. \quad (4)$$

The system can be modeled by a synchronous MSFG in Figure 6, where

$$\mathbf{P}_1 = \begin{bmatrix} 3 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 2 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 2 \end{bmatrix}, \quad \mathbf{P}_2 = \begin{bmatrix} 2 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 3 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 1 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 1 \end{bmatrix},$$

$$\mathbf{P}_3 = \begin{bmatrix} 1 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 4 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 2 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 1 \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ e & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & e & \varepsilon \end{bmatrix}, \quad (5)$$

$$\mathbf{J} = \begin{bmatrix} e \\ \varepsilon \\ \varepsilon \\ \varepsilon \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & e \end{bmatrix}.$$

Applying the return loop method to find the gain of the graph from $\mathbf{u}_f$ to $\mathbf{x}_f$ we have

$$\mathbf{T}_{\mathbf{x}_f,\mathbf{u}_f} = (\mathbf{P}_3(\mathbf{P}_2(\mathbf{P}_1\mathbf{H})^\star\mathbf{H})^\star\mathbf{H})^\star\mathbf{P}_3 \otimes (\mathbf{P}_2(\mathbf{P}_1\mathbf{H})^\star\mathbf{H})^\star\mathbf{P}_2(\mathbf{P}_1\mathbf{H})^\star\mathbf{P}_1.$$

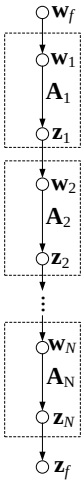

Fig. 7. Alternative way to model a permutation flow shop.

Suppose that all the machines are initially available, i.e. $\mathbf{w}_f = [\varepsilon\ \varepsilon\ \varepsilon\ \varepsilon]^T$ and that all parts are available to the system at time zero, i.e. $\mathbf{u}_f = [e\ e\ e\ e]^T$. Then

$$\mathbf{x}_f = \mathbf{T}_{\mathbf{x}_f,\mathbf{u}_f}\mathbf{u}_f = \begin{bmatrix} 6 & \varepsilon & \varepsilon & \varepsilon \\ 12 & 9 & \varepsilon & \varepsilon \\ 14 & 11 & 4 & \varepsilon \\ 15 & 12 & 5 & 4 \end{bmatrix} \begin{bmatrix} e \\ e \\ e \\ e \end{bmatrix} = \begin{bmatrix} 6 \\ 12 \\ 14 \\ 15 \end{bmatrix}.$$

Therefore, the system's makespan equals 15 (time units).

In the modeling method described above, we first obtained synchronous MSFG models of each machine and then stacked these synchronous MSFGs horizontally to obtain the model of the flow shop. Alternatively, the system can be modeled from a different perspective. Figure 7 illustrates the idea. The system is modeled as a set of synchronous MSFGs stacked vertically, where each synchronous MSFG represents a manufacturing process of a part processed by a set of machines (e.g. Figure 4).

Example Consider the flow shop defined in the previous example. It can be modeled by the synchronous MSFG shown in Figure 7, where

$$\mathbf{A}_1 = \begin{bmatrix} t_{1,1} & e & \varepsilon \\ t_{1,1}t_{2,1} & t_{2,1} & e \\ t_{1,1}t_{2,1}t_{3,1} & t_{2,1}t_{3,1} & t_{3,1} \end{bmatrix} = \begin{bmatrix} 3 & e & \varepsilon \\ 5 & 2 & e \\ 6 & 3 & 1 \end{bmatrix},$$

$$\mathbf{A}_2 = \begin{bmatrix} 2 & e & \varepsilon \\ 5 & 3 & e \\ 9 & 7 & 4 \end{bmatrix}, \mathbf{A}_3 = \begin{bmatrix} 1 & e & \varepsilon \\ 2 & 1 & e \\ 4 & 3 & 2 \end{bmatrix}, \mathbf{A}_4 = \begin{bmatrix} 1 & e & \varepsilon \\ 3 & 1 & e \\ 4 & 2 & 1 \end{bmatrix}.$$

We have

$$\mathbf{T}_{z_f,w_f} = \mathbf{A}_4\mathbf{A}_3\mathbf{A}_2\mathbf{A}_1 = \begin{bmatrix} 12 & 9 & 7 \\ 14 & 11 & 9 \\ 15 & 12 & 10 \end{bmatrix}$$

and $\mathbf{z}_f = \mathbf{T}_{z_f,w_f}\mathbf{w}_f$ Assume that the system's input $\mathbf{w}_f = [e\ e\ e]^T$, *i.e.* all the machines are available to the system at time zero. Then,

$$\mathbf{z}_f = \begin{bmatrix} 12 \\ 14 \\ 15 \end{bmatrix}$$

The makespan of the system is 15 time units.

The modeling method illustrated in Figure 6 is referred to as *machine-based modeling approach*, because we are essentially stacking machines horizontally. The other modeling method shown in Figure 7 is referred to as *part-based modeling approach*, because it essentially involves stacking parts vertically. Both modeling approaches can be used to analyze the system. In fact they both give the same result for the makespan (15 time units) for the example presented in this section. However one approach may be more efficient computationally depending on the application. For example, a part-based approach can be used for scheduling applications because changing the order of parts on a machine, simply corresponds to swapping order of blocks in the vertical stack of blocks in Figure 7. A machine-based modeling approach, on the other hand, can be used for buffer allocation problems, because adding a buffer between the machines amounts to inserting a unit capacity buffer (represented by a SFG shown in Figure 3(b)) into the horizontal stack of machines in Figure 6.

VII. Conclusion

Synchronous MSFGs are MSFGs over max-plus algebra. The paper has presented the theory and applications of synchronous MSFGs. New topological methods for evaluating gains of synchronous MSFGs have been described. It has been shown that $\mathbf{A}^*$ in max-plus algebra is equivalent to $(\mathbf{I}-\mathbf{A})^{-1}$ in regular algebra. This observation allowed us to extend the existing theory for regular MSFGs to synchronous MSFGs. An application of synchronous MSFGs to modeling and performance evaluation of manufacturing systems has been illustrated.

References

[1] S.J. Mason. Feedback theory–some properties of signal-flow graphs. In *Proc. Institute of Radio Engineers*, volume 41, pages 1144–1156, Sept. 1953.

[2] J.O. Pliam. An algebraic approach to signal flow graph theory. Master's thesis, University of Minnesota, May 1992.

[3] D.E. Riegle. *Topological properties of matrix signal flow graphs*. PhD thesis, Purdue University, June 1971.

[4] D.E. Riegle and P.M. Lin. Matrix signal flow graphs and an optimum topological method for evaluating their gains. *IEEE Transactions on Circuit Theory*, CT-19(5):427–435, 1972.

[5] B. Hruz and M.C. Zhou. *Modeling and Control of Discrete-event Dynamic Systems with Petri Nets and Other Tool*. Springer-Verlag, London, UK, 2007.

[6] Christos G. Cassandras and Stephane Lafortune. *Introduction to Discrete Event Systems*. Springer US, Boston, MA, USA, 2007.

[7] G. Cohen, S. Gaubert, and J. Quadrat. Max-plus algebra and system theory: Where we are and where to go now. *Elsevier Annu. Rev. Control*, 23:207–219, 1999.

[8] F. Baccelli, G. Cohen, G.J. Olsder, and J.P. Quadrat. *Synchronization and Linearity*. John Wiley and Sons, West Sussex, England, 1992.

[9] A. Imaev and R.P. Judd. Block diagram-based modeling of manufacturing systems using max-plus algebra. In *Proceedings of the American Control Conference*, June 10 - Jun 12 2009.

[10] A. Imaev. *Hierarchical modeling of manufacturing systems using max-plus algebra*. PhD thesis, Ohio University, 2009.

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

[12] J.O. Pliam. Ring graphs and gain formulas, an algebraic approach to topology. In *IEEE International Symposium on Circuits and Systems*, pages 327–330, May 1989.

[13] S. Thomas McCormick, M. Pinedo, Scott Shenker, and Barry Wolf. Sequencing in an assembly line with blocking to minimize cycle time. *Oper. Res.*, 37(6):925–935, 1989.