Design of An FMS Controller for Job-Shop Scheduling and Production Control

Hsiang-Hsi Huang†, E. Del Castillo‡, and F.L. Lewis†

†Automation & Robotics Research Institute, the University of Texas at Arlington, 7300 Jack Newell Blvd. S., Ft. Worth, TX 76118-7115, USA. Tel: (817) 794-5934; Fax: (817) 794-5952. E-mail: hhuang@arri.uta.edu; flewis@arri.uta.edu.
‡Industrial & Manufacturing Systems Engineering Dept, the University of Texas at Arlington, Box 19017, Arlington, TX 76019-0017, USA. Tel: (817) 273-3220; Fax: (817) 273-3406. E-mail: D301EDC@UTARLVM1.UTA.EDU.

Abstract

A matrix-based formulation for FMS controller design for job-shop manufacturing system is proposed using standard manufacturing tools such as the BOM, task sequencing matrix, and resource requirements matrix. The proposed matrix formulation overcomes the major difficulties involved in using Petri nets for FMS scheduling control, and allows design for deadlock removal and conflict resolution. In the formulation, the controller is comprised of inner loops where no shared resources are involved, and outer share-resource and route-selection loops that require some conflict-resolution decision making.

The paper presents the controller formulation for a job-shop manufacturing system where parts may have alternate routings. Dispatching rules are used to resolve conflicts in scheduling and routing decisions, which are treated as dual problems. A hierarchical scheduling system to manage manufacturing systems results from the development of this mathematical formulation. The integration of the discrete event FMS controller with upper level production control systems, such as an MRP system, are discussed.

1. Introduction

In this paper, a systems theory point of view is used to develop a new matrix description that provides a rigorous, repeatable, design algorithm and analysis tools for FMS controllers, and gives a basis for conflict resolution. Matrix equations are used over a nonstandard ("and","or") algebra to describe the system structure, where the description of a new system with different part routings can quickly be obtained by changing only an *operation sequencing matrix* S_u. The controller is comprised of outer and inner loops, where the inner loop is decision-free with no shared resource conflicts. The outer loop, however, involves shared resources and route selection and requires a conflict resolution command input and a route decision input, u_C and u_R, selected based on dispatching rules. The theory unites IE tools such as the bill of materials (BOM) [1,3], resource requirements matrix [8] and design sequencing matrix [4,14,15] with Petri net (PN) and max/plus formulations [2,12]. FMS problems recently studied in the literature such as deadlock avoidance [17] can be analyzed with the matrix controller formulation [13]. The formulation is presented for the general case of a job shop with alternate routings.

This paper is organized as follows: In Section 2 we introduce different job-shop configurations accommodated by the matrix formulation. In Section 3 we present the FMS controller matrix formulation. Different building structures are presented, from the simplest case of a single-machine-single-part (SMSP) system, up to the multiple-machine-multiple-part (MMMP) controller formulation. The matrix formulation for the controller of job-shop with alternate routings is presented. It is shown how the FMS controller matrix equations directly yield the PN description. Finally Section 4 shows how the proposed FMS controller can be integrated with upper-level production control systems, such as an MRP system. The hierarchical scheduling system as the decision structure of manufacturing systems then can be constructed. The overall system structure with production control is shown in Fig. 1.

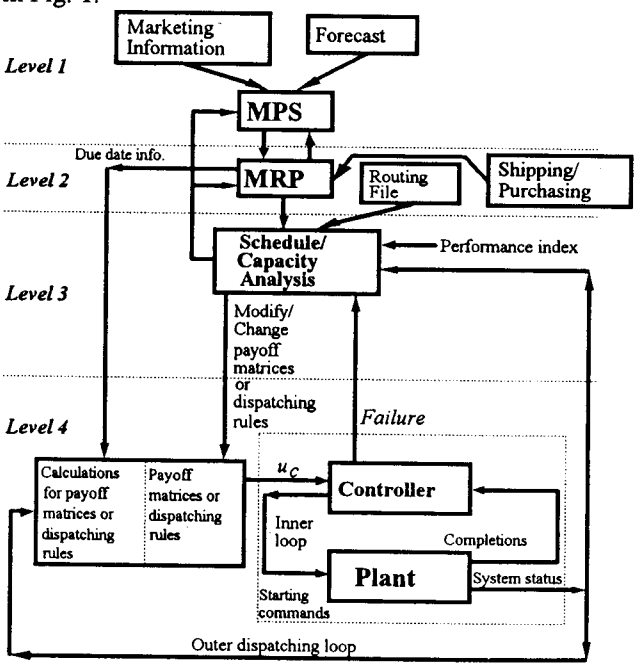

Fig. 1 A four-level system structure of production control.

2. Job-Shop Scheduling Models

Traditionally, a job-shop production system has been defined in the production engineering literature [3] as an intermittent or batch-oriented production system with a functional layout. We can identify two basic types of job-

shops, namely, the *job-shop with fixed part routing* and the *job-shop with alternate part routings.*

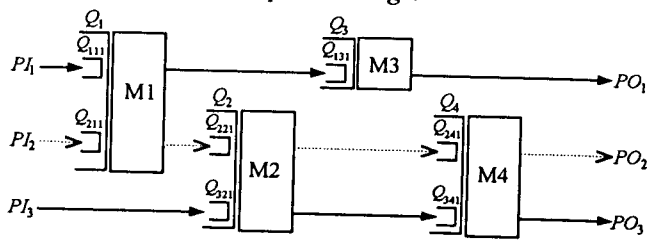

Fig. 2a Job-shop with fixed routing without reentrant flow.

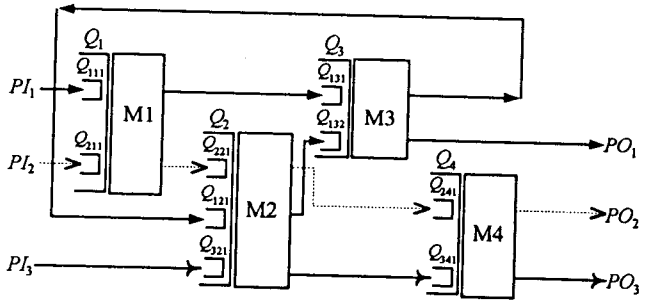

Fig. 2b Job-shop with reentrant flow.

2.1. Job-Shop with Fixed Routing

A job-shop design basically refers to a production system structure where different parts (or jobs) are processed through different machines according to multiple sequences or routings. We refer to the case of multiple parts forming multiple sequences as a job-shop with fixed part routing. In this case, the routing of each part type is fixed (e.g. see Fig. 2a). An extension of this is the case of reentrant flow of parts, i.e. the same part can re-enter the same machine more than once for different operations but the routing of each part is fixed (see Fig. 2b). If multiple routes exist for one or more parts we use the term *job-shop with alternate routings.*

When multiple parts are manufactured in a workcell, there exists a potential conflict arising whenever resources are shared. In the shared-resource problem, several parts simultaneously request the same resource for operation. To resolve shared-resource conflicts, an external input u_C needs to be introduced to schedule the tasks. We call u_C the conflict resolution input. It is selected based on dispatching rules that will activate only one operation or task from the conflicting set.

2.2. Job-Shop with Alternate Routings

A complex job-shop may have variable or alternate routings. This means that in addition to the shared-resource conflict problem, there is a possibility of task choice or route selection with respect to machines as illustrated in Fig. 3. To solve the choice arising from route selection, another external decision input defined as the route decision input, u_R, should be added to resolve the route selection. The input u_R is also based upon a group of dispatching rules with respect to some specific performance index and selects a route. As we discuss in Section 4, routing and scheduling can be considered, in a sense, as dual problems (see Fig. 4).

Flexibility in job routing is a key current issue in FMS design, and suitable rules for job routing selection are not yet well developed.

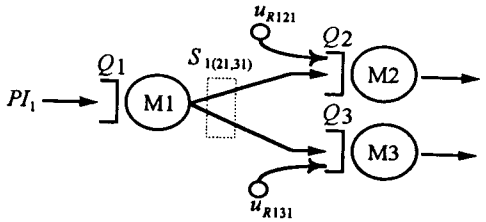

Fig. 3 The basic alternate routings with u_R the route decision input.

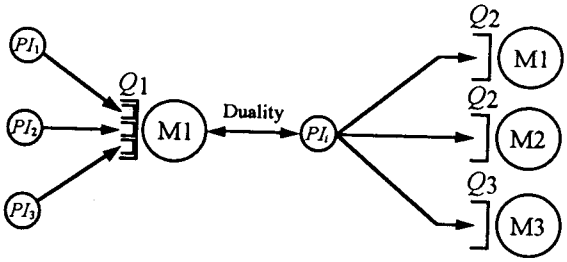

Fig. 4 Duality between the part and route selection.

3. Matrix Representation of Workcell Structure

In order to design a manufacturing system, a few basic manufacturing cell structures may be constructed. By using these basic workcell structures as building blocks, a matrix framework for describing a manufacturing system can be developed and implemented efficiently. In the rule-based controller formulation described in references [10,12] a matrix "and"/"or" algebra is used, where standard matrix operations of "multiplication" and "addition" are replaced respectively by logical "and" and "or" operations, and status vectors are expressed in *negative logic* (i.e., "0" represents task completion and task start commands).

In the following discussion, four basic manufacturing cell structures will be introduced step by step. One can obtain PN descriptions using this step-by-step design procedure which is based upon standard industrial engineering approaches. Using these four basic structures, one can easily describe the behavior and formulate a scheduling controller for many different manufacturing systems.

3.1. Single Machine for Single and Multiple Parts (SMSP and SMMP)

3.1.1. The single-machine-single-part case (SMSP):

Consider a manufacturing *unit* shown as Fig. 5a&5b, in which a part $P1$ is inputted (denotes as PI) to a queue $Q1$ before processing on machine $M1$ (operation O). PP represents a part presents in the workcell and PO denotes the part is finished and then outputted to warehouse. The corresponding PN structure is illustrated in Fig. 5c by defining O_{ijk} as the operation for part type i processed on machine j the k-th time. We assume the machine is loaded and unloaded automatically.

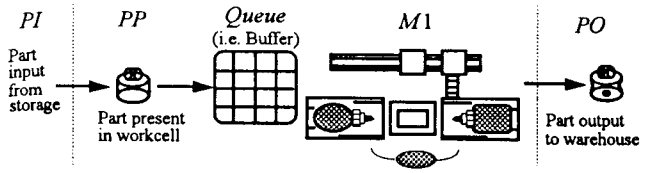

Fig. 5a The product flow of a manufacturing unit.

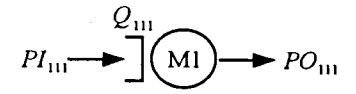

Fig. 5b The simplified representation of Fig. 5a.

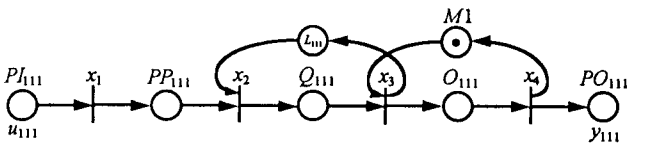

Fig. 5c The Petri net corresponding to the SMSP case.

Denote by $V = \{PP_{111}, Q_{111}, O_{111}\}$ the set of all the operations to be performed, $u_{111} = \{PI_{111}\}$ the part input operation, and $y_{111} = \{PO_{111}\}$ the part output operation. The set of the resources is denoted by $R = \{L_{111}, M1\}$. From a systems theory perspective, we define two sets of logical variables in *negative logic* with the same name as those in V and R, superscripted respectively by c and s, where c signifies the completion of an operation or resource release, and s represents start of an operation or resource release (logical values 1 for false, 0 for true in negative logic).

By applying a set of *if-then-else* rules, the task sequencing combining with resource requirements can be described as follows:

if $(PI^c_{111} = 0)$ then $(PI^s_{111} = 0)$ else $(PI^s_{111} = 1)$

if $(PP^c_{111} = 0$ and $L^c_{111} = 0)$ then $(Q^s_{111} = 0)$ else $(Q^s_{111} = 1)$

if $(Q^c_{111} = 0$ and $M1^c = 0)$ then $(O^c_{111} = 0)$ else $(O^s_{111} = 1)$

if $(O^c_{111} = 0)$ then $(PO^s_{111} = 0$ and $M1^s = 0)$ else

$(PO^s_{111} = 1$ and $M1^s = 1)$. (1)

As one can see, PP^s_{111}, Q^s_{111}, O^s_{111}, PO^s_{111} are equal to 1 or 0, depending on the completion condition of the precedent process(es). These contain the information needed for sequencing tasks, and *controller state variables* can be introduced as

$$x_1 = PI^c_{111},\ x_2 = PP^c_{111},\ x_3 = Q^c_{111},\ x_4 = O^c_{111}. \quad (2)$$

The controller sets the corresponding c variable to zero immediately after a condition $x_i = 1$ (i=1-4) occurs. By using the symbols "•" and "+" with the meaning of logical "and" and "or" in *negative logic* and combining the task sequencing and the resource requirements, the *controller state equation* can easily be written as

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} PP^c_{111} \\ Q^c_{111} \\ O^c_{111} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \cdot L^c_{111} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \cdot M1^c + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \cdot PI^c_{111}. \quad (3)$$

The matrix form of the rules for starting task operation and starting resource release is expressed by

$$\begin{bmatrix} PP^s_{111} \\ Q^s_{111} \\ O^s_{111} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}, \quad (4)$$

and $$L^s_{111} = [0\ \ 0\ \ 1\ \ 0] \cdot [x_1\ \ x_2\ \ x_3\ \ x_4]^T,$$

$$M1^s = [0\ \ 0\ \ 0\ \ 1] \cdot [x_1\ \ x_2\ \ x_3\ \ x_4]^T. \quad (5)$$

The output equation is:

$$PO^s_{111} = y_{111} = [0\ \ 0\ \ 0\ \ 1] \cdot [x_1\ \ x_2\ \ x_3\ \ x_4]^T. \quad (6)$$

The rule-based controller structure for equations (3) to (6) is shown in Fig. 6 and consists of the following equations [12]:

- **Controller state equation:**

$$x_{p1} = F_{v111} \cdot v^c_{111} + F_{L111} \cdot r^c_{L111} + F_{M111} \cdot r^c_{M1} + F_{u111} \cdot u_{111} \quad (7)$$

- **Task start equation:** $v^s_{111} = S_{v111} \cdot x_{p1}$ (8)

- **Resource release equation:** $r^s_{L111} = S_{L111} \cdot x_{p1}$

$$r^s_{M1} = S_{M111} \cdot x_{p1} \quad (9)$$

- **Output equation:** $y_{111} = S_{y111} \cdot x_{p1}$. (10)

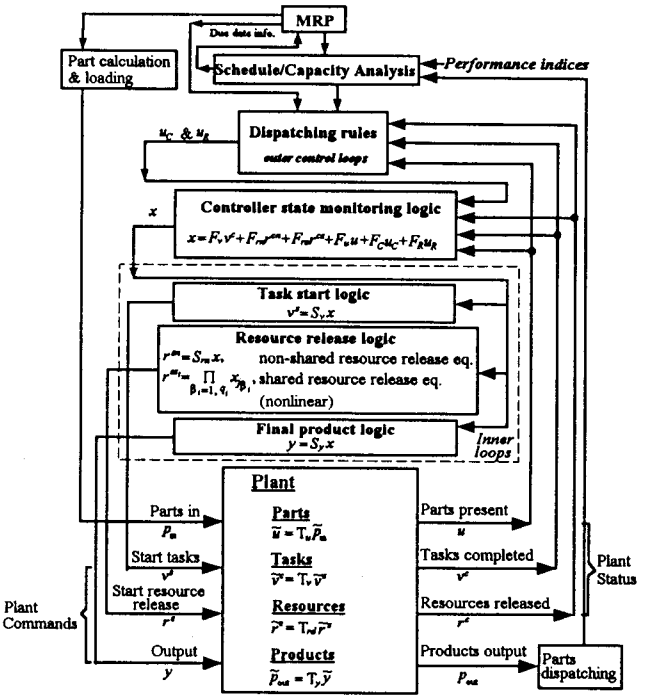

Fig. 6 The rule-based FMS controller for a manufacturing system with production control integration.

Here, x_{p1} is the controller state vector for part type P1, $v^c_{111}(v^s_{111})$ is the task completion (start) vector, r^c_{L111} and r^c_{M1} (r^s_{L111} and r^s_{M1}) are the resource completion (release) vectors, u_{111} and y_{111} are the input and output vectors respectively. As one can see later, one part type might have to go through different operation processes. This means that the controller state vector x_{pi}, i=1, 2,..., can contain sub-vectors for different workcells. For clarity, x_{pi} (the controller state vector of part type i) is now defined as the *controller state vector of the machine unit*, and denoted by x_m.

In this formulation, F_{v111} is called the task sequencing matrix, F_{L111} and F_{M111} are called the resource requirement

matrices, and F_{v111} the input matrix. S_{v111} is called the task start matrix, S_{L111} and S_{M111} are called the resource release matrices, and S_{y111} is called the output matrix. From the controller state equation (7), it is important to mention that F_{v111} corresponds to Steward's precedence (sequencing) matrix [14] and F_{L111}, F_{M111} are equivalent to Kusiak's resource requirements matrices [8].

These developments show how to obtain a matrix description of the single-machine-single-part-type (SMSP) case using a step-by-step design procedure. For SMSP there are no choice tasks or shared resources. The SMSP case forms the basic element of every manufacturing system structure. From the SMSP structure representation, one can construct different controllers for different specific manufacturing systems as revealed below.

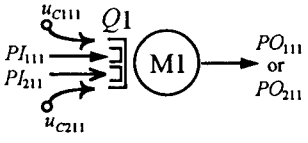

Fig. 7a The simplified representation of the SMMP case.

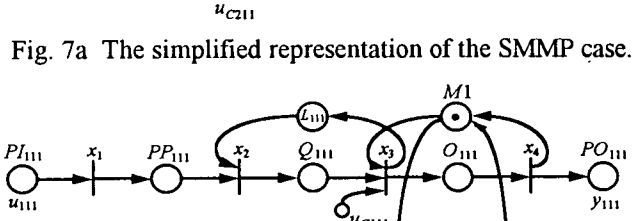

Fig. 7b The Petri net corresponding to the SMMP case.

3.1.2 The single-machine-multiple-part case (SMMP):

The SMMP structure can be described as several similar yet separated processing paths through the same machine by applying the SMSP representation as the basic block structure. Using the Fig. 7a&7b to illustrate the multiple processes through the same machine, i.e. $M1$, the controller state equation and the matrices for starting task and resource release can be defined as the block structure for the two parts case shown as follows:

• Controller state equation:

$$x = x_p = \begin{bmatrix} x_{p1} \\ x_{p2} \end{bmatrix} \overset{\Delta}{=} \begin{bmatrix} x_{m111} \\ x_{m211} \end{bmatrix} = \begin{bmatrix} F_{v111} & 0 \\ 0 & F_{v211} \end{bmatrix} \cdot \begin{bmatrix} v^c_{111} \\ v^c_{211} \end{bmatrix} + \begin{bmatrix} F_{L111} & 0 \\ 0 & F_{L211} \end{bmatrix} \cdot \begin{bmatrix} r^c_{L111} \\ r^c_{L211} \end{bmatrix} +$$

$$\begin{bmatrix} F_{M111} & 0 \\ 0 & F_{M211} \end{bmatrix} \cdot r^c_{M1} + \begin{bmatrix} F_{u111} & 0 \\ 0 & F_{u211} \end{bmatrix} \cdot \begin{bmatrix} u_{111} \\ u_{211} \end{bmatrix} + \begin{bmatrix} F_{C111} & 0 \\ 0 & F_{C211} \end{bmatrix} \cdot \begin{bmatrix} u_{C111} \\ u_{C211} \end{bmatrix}$$

$$\overset{\Delta}{=} F_v v^c + F_L r^c_L + F_{M1} r^c_{M1} + F_u u + F_C u_C, \qquad (11)$$

where x_p is the augmented controller state vector for both part types.

• Task start equation:

$$v^s = \begin{bmatrix} v^s_{111} \\ v^s_{211} \end{bmatrix} = \begin{bmatrix} S_{v111} & 0 \\ 0 & S_{v211} \end{bmatrix} \cdot \begin{bmatrix} x_{p1} \\ x_{p2} \end{bmatrix} \overset{\Delta}{=} S_v x_p \qquad (12)$$

• Non-shared resource release equation:

$$r^{sn}_L = \begin{bmatrix} r^s_{L111} \\ r^s_{L211} \end{bmatrix} = \begin{bmatrix} S_{L111} & 0 \\ 0 & S_{L211} \end{bmatrix} \cdot \begin{bmatrix} x_{p1} \\ x_{p2} \end{bmatrix} \overset{\Delta}{=} S_L x_p \qquad (13)$$

where r^{sn}_L is the non-shared resource release vector and S_L is the resource release matrix.

• Shared resource release equation (nonlinear):

$$r^{ss}_{M1} = x_4 \cdot x_8 \qquad (14)$$

where r^{ss}_{M1} is the shared-resource release vector.

• Output equation: $y = \begin{bmatrix} S_{y111} & S_{y211} \end{bmatrix} \cdot \begin{bmatrix} x_{p1} \\ x_{p2} \end{bmatrix} = S_y x_p.$ (15)

By employing the representation of block matrix structure shown above, different specific manufacturing systems, e.g. the multiple-machine-multiple-part (MMMP) case discussed later, can be directly constructed.

When a single machine can be used for processing multiple part types, an external shared-resource input is necessary to resolve the conflict arising due to the possibility of the presence of several parts simultaneously. Therefore, we introduce u_C the conflict resolution input in the rule-based controller structure as shown in Fig. 7b [6]. The resources are now classified into non-shared (e.g., L_{111} & L_{211}) and shared (e.g., $M1$) resources. The u_C command is selected using the dispatching rules based on selected performance indices or policies (see the dispatching loop in Fig. 6).

It is worth mentioning that in the shared resource case, the only difference in the controller equations is the nonlinear form of the shared-resource release equation. For instance, in equation (14) the shared-resource release vector for Fig. 7b is $r^{ss}_{M1} = x_4 \cdot x_8$, i.e. $r^{ss}_{M1} = x_4$ .or. x_8. This shows the potential conflict arising from simultaneous requests to perform operations O_{111} and O_{211} on the same machine $M1$. To solve such conflict arising from sharing the same resource, the outer loop term $F_C u_C$ as required by equation (11) to obtain the controller state equation is introduced, where $u_C=[u_{C111}\ \ u_{C211}]^T$ is the conflict resolution input of the example of Fig. 7b (see Fig. 6).

A deadlock analysis can be performed after the rule-based controller has been designed as shown by the next result.

Theorem 1: One can avoid deadlock by either:
1) adding more resources to the shared resource pool $M1$
2) adjusting the number of pallets (i.e. limiting the number of $P1$ and $P2$)
3) modifying the antecedents of the synchronization rules (i.e. the structure of matrices F_{M111}, F_{M211})
4) appropriate selection of u_C

The proof, including the qualitative and quantitative discussion, can be found in [12]. ■

By applying Theorem 1 we can obtain a conflict-free manufacturing system, i.e. decision-free, and then the max/plus representation [2] can be computed from the linear controller equations [12]. Once the rule-based controller equations are rewritten in max/plus terms, a workcell performance analysis can be efficiently carried out.

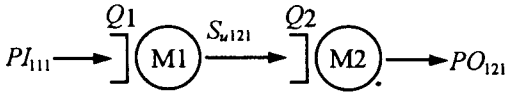

Fig. 8 The representation of the MMSP case.

3.2. The Manufacturing System Design

3.2.1. The multiple-machine-single-part case (MMSP):

When a single part type goes through multiple machines, two basic flow structures can be constructed. One of these structures is the *flow-line* structure shown in Fig. 8. The second structure is the *route-selection* or *choice* structure as illustrated in Fig. 3. The next discussion shows important properties of these two basic structures for manufacturing system design.

***Flow-line structure*:** When a single part goes through a series of machines, the *operation sequencing matrix* S_u needs to be introduced to connect consecutive processes. The S_u matrix is a reusable module and is responsible for connecting every two consecutive machine units in the order in which the parts visit the machines as Fig. 8 shows. By only changing S_u, a new system can be obtained [6]. Define $S_{i(jk\rightarrow lm)}$ as the *closed-loop sequencing matrix*. Then the interconnection of the consecutive machine units can be described by

$$S_{i(jk\rightarrow lm)} = F_{uilm} S_{uilm} S_{yijk},$$

where $i(jk\rightarrow lm)$ stands for part type i processed by machine j the k-th time proceeding to machine l for the m-th time.

Here, by using information on which input parts are needed for which tasks, the part input vector u is defined as a vector with elements corresponding to all the parts that enter as raw materials into the workcell. u_{ilm} stands for the input vector of part i processed by machine l the m-th time. F_{uilm} is the part input matrix with respect to u_{ilm}. In F_{uilm}, all elements are 0, except for 1's included in positions (i, j) if entering part i is a needed for task j. S_{uilm} denotes the operation sequencing matrix in which all elements are 0, except for 1's representing part i will be processed by machine l the m-th time. The S_{uilm} matrix shows the condition of connecting two consecutive machines. In a flow line structure, the S_{uilm} matrix always equals 1 for two consecutive machines. This means that the next machine for processing part i the m-th time will always be machine l. From the information on which products result from which tasks, we define the product output vector y as a vector with elements corresponding to all the finished products. y_{ijk} is the product output vector with the meaning of the output of part i processed by machine j the k-th time. By using the closed-loop sequencing matrix $S_{i(jk\rightarrow lm)}$ as defined, the general formation of connection of two consecutive machines can be obtained [6].

***Route- selection structure*:** For the route selection case, i.e. with alternate routings, there may be a need for selecting the next machine where the operation of the part takes place (see Fig. 3). When there is route selection, a *route decision input* u_R needs to be introduced to resolve such route choosing problems. With the input of u_R, only one route can be activated to produce the final product.

Denote $S_{i(jk,lm)}$ the *choice-operation sequencing* vector, then the interconnection of the consecutive machine units with alternate routings can be described by

$$S_{i(jk,lm)} = [S_{uijk} \; S_{uilm}]^T,$$

where $i(jk,lm)$ denotes that part type i in the current machine can be processed next either by machine j the k-th time or by machine l the m-th time. For the alternate routings, there are multiple 1s in the same column of the choice-operation sequencing vector. Using matrix notation, the representation of the input matrix with respect to u_R can be obtained by the following simple transformation

$$\begin{aligned} &[x_h \; x_q]^T = I\,[u_{Rijk} \; u_{Rilm}]^T, \\ &[u_{ijk} \; u_{ilm}]^T = I^2\,[u_{Rijk} \; u_{Rilm}]^T = [u_{Rijk} \; u_{Rilm}]^T. \end{aligned} \qquad (16)$$

where I is the identity matrix, and $[x_h \; x_q]^T$ denotes the vector containing transitions with choice tasks. $[u_{ijk} \; u_{ilm}]^T$ denotes the input vector and $[u_{Rijk} \; u_{Rilm}]^T$ is the route decision input vector. Since $[u_{Rijk} \; u_{Rilm}]^T = [u_{ijk} \; u_{ilm}]^T$, the *route-selection input matrix* F_R with respect to the route decision input u_R can be represented by the input matrix F_u This interpretation shows that $F_R = [F_{Rijk} \; F_{Rilm}]^T$ is the same as $F_u = [F_{uijk} \; F_{uilm}]^T$ [6]. It is very interesting to mention that when there exists only one 1 in $S_{i(jk,lm)}$ the choice-task sequencing vector, the structure becomes the flow line structure.

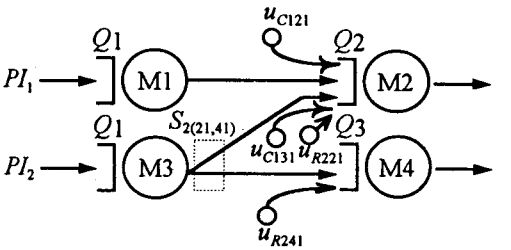

Fig. 9 The representation of the MMMP case.

3.2.2. The multiple-machine-multiple-part case (MMMP):

A multiple-machine-multiple-part type structure (MMMP) that describes a general manufacturing system can now be obtained by combining all the basic cases just mentioned (e.g. see Fig. 9). The MMMP job-shop design problem contains both shared resource and task choice problems which need external decision inputs to resolve resource conflict and route selection. With the use of a block matrix structure, one can combine the basic cases to obtain a matrix description of a general manufacturing system.

3.2.3. Petri Net from matrix controller equations

An important property of the matrix formation is illustrated by the next result, which shows that a PN description can be directly obtained from it. This provides the first repeatable design algorithm for Petri nets the authors are aware of.

Define $F_v=[F_{vn} \; F_{vc}]$ the task sequencing matrix, where the blocks F_{vn}, F_{vc} correspond to non-choice and choice-tasks; $F_r= [F_{rn} \; F_{rs}]$ the resource requirement matrix, where the blocks F_{rn}, F_{rs} correspond to non-shared and shared-resources respectively. In the SMSP case above, one can see that $F_{v111} \in F_{vn}$ and F_{L111}, $F_{M111} \in F_{rn}$.

Define the task start matrix, where the blocks S_{vn}, S_{vc} correspond to non-choice and choice-tasks; the resource release matrix, where the blocks S_{rn}, S_{rs} correspond to non-shared and shared-resources, respectively. As one can see in the SMSP case, $S_{v111} \in S_{vn}$ and S_{L111}, $S_{M111} \in S_{rn}$.

Theorem 2: The rule-based FMS controller with incidence matrices $W^- = F = [F_v\ F_r]$ and $W^+ = S^T = [S_v^T\ S_r^T]$ is equivalent to a Petri Net. Proof is shown in [12]. ■

4. Integration with an MRP System

The flexibility of a manufacturing system is related to the ability of the manufacturing company to produce different products in the most efficient way; the efficiency of a manufacturing system relies on the effective planning and control of day-to-day operations. Of critical importance is the scheduling of production operations and the control of planned activities [16]. On-time delivery of products and customer satisfaction are the goals that every company strives for.

The scheduling of a manufacturing system can be split into four different levels as shown in Fig. 1. The first level of the system is considered to be the highest level of decision making; the overall schedule of the organization, or static schedule, known as Production Plan (PP) is developed through information provided by the marketing department forecasts and upper management plans [1,3,9]. A midterm schedule known as the Master Production Schedule (MPS) is formed after the production plan has been established. Next, a tentative shipping/purchasing schedule of material is acquired using scheduling tools such as Material Requirements Planning (MRP). The third level selects the decision-making dispatching rule for the system based on internal and external factors and performance measures. In the fourth level, a given dispatching rule is executed on the shop floor until further notice. Schedulers have been using many different rules in order to reach a suboptimal schedule which is implementable by the company [8,11].

4.1. MRP Calculations

MRP is a computational technique that converts the MPS into a detailed schedule for raw materials and components used in the end products. Using the order releases generated by the MRP system, the FMS controller can be integrated with upper levels of the manufacturing system. Parts are loaded into the FMS workcell based on minimizing the difference between the actual products produced and the scheduled orders released. This control concept is also used to dispatch parts to machines within the workcell whenever necessary.

The well-known MRP algorithm is given as follows (see, for example, [1,3]). The formulas give the schedule of material required at each level of the BOM and incorporate the backtracking used in MRP:

Define the planning horizon as t = 1, 2, ..., T; i is the part type or product.

Let $W_{i,t} = x_{i,t-1} + Q_{i,t} - U_{i,t}$, where: 1) $x_{i,t}$ is the Final Inventory of period t, 2) $Q_{i,t}$ is the Scheduled Order receipt of period t, 3) $U_{i,t}$ is the Gross Requirement of period t;

$x_{i,t} = (W_{i,t})^+$, where $(\cdot)^+$ denotes positive part,

$y_{i,t} = -(W_{i,t})^-$, where $y_{i,t}$ = Net Requirement; $(\cdot)^-$ denotes negative part,

$Q_{i,t} = y_{i,t}$,

$Z_{i,t-L_i} = y_{i,t}$, where: 1) L_i is the Lead Time of part i, 2) is the Scheduled Order Release of period t,

$U_{i,t} = \sum_{\substack{\text{all predecessors} \\ k}} Z_{k,t} \times b_{i,k} + d_{i,t}$, where: 1) $b_{i,k}$ is the (i,k) element of the BOM matrix, 2) $d_{i,t}$ is the Independent Demand (if any).

4.2. FMS Dispatching Rules

There exist a wide variety of dispatching rules that can be classified in many different ways [10,11]. A convenient classification for our purposes is *buffer-based* dispatching rules and *part/machine-based* dispatching rules. In the latter case, dispatching is performed according to a part or machine attribute. For example, SI (shortest imminent operation time) dispatches based on T_{ij}, the processing time of part i in machine j. Buffer-based dispatching rules assign priorities according to an attribute related to the buffers or queues of the machine. For example, LBFS (last buffer-first served) gives higher priority to those parts that are visiting the machine for the last time (i.e., they are in the last buffer of the machine).

In our controller formulation, dispatching rules are utilized to resolve two types of conflicts. The first type of conflict is *scheduling*, when a conflict resolution input u_C is needed to select one task among many tasks that simultaneously request the same resource. The second type of conflict is *routing*, when a conflict resolution input u_R is needed to select one machine (next step in the route) among many alternative routes that a part may follow.

Table 4.1. Common FMS Dispatching Rules

Buffer		**Part/Machine**	
Routing	Scheduling	Routing	Scheduling
FBFS (First Buffer First Served)	**LBFS** (Last Buffer First Served)	**SI** (Shortest Imminent time)	**SR** (Shortest Remaining time)
SNQ (Shortest Non-full Queue)	**SRC** (Shortest Remaining Capacity)	**LI** (Largest Imminent time)	**LR** (Largest Remaining time)
Random	**Random**	**FIFO** (First In First Out)	**FIFO** (First In First Out)
			EDD (Earliest Due Date)

It is interesting to note that the routing and scheduling problems are *dual*. Duality in this case is observed by interchanging the role of parts and machines and reversing the flow of parts (see Fig. 4). In addition, we can form pairs of *dual dispatching rules* if the attributes they are

based on are similar but one rule is used for scheduling whereas its dual is used for routing purposes. Table 4.1 contains the most common dispatching rules used in practice classified according to buffer or part/machine and showing some dualities. Not all dispatching rules have dual counterparts. For instance, EDD (earliest due date) is usually applied in scheduling problems but does not have a dual for routing purposes. This tends to happen when the attribute of the rule (e.g. due date) depends on the part but not on the machine.

4.3. Hierarchical Scheduling System (HSS)

To manage manufacturing systems, dispatching rules will assist the schedulers in making a specific conflict resolution decision. Our formulation reveals a hierarchical scheduling system to have the four-loop manufacturing production system structure shown in Fig. 10. We will refer to these four loops as *part loading*, *router*, *dispatcher* and *decision-free* loop.

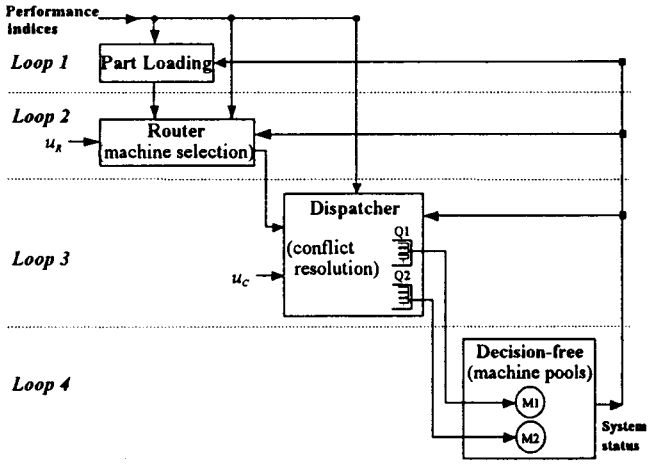

Fig. 10 A four-loop hierarchical scheduling structure.

Based on the production planning information produced by the MRP system, a dynamic part loading rule is used to select the part type to load into the shop floor from the warehouse. After parts are loaded into the manufacturing workcell, the router containing the part routing information is responsible for dispatching parts to machines at the second loop of the hierarchy. It is worth mentioning that the router information is contained exactly in the operation sequencing matrix, S_u. A new route is obtained after changing the S_u matrix, or equivalently, by changing the route information of the router. In the alternate routings problem, the route decision input, or u_R vector, solves such task choice situations by selecting only one of the choice requests made. The third loop is the dispatcher where a dispatching decision input u_C is made for resolving the shared-resource problem. After conflicts have been resolved, a decision-free system is gained and products are directly processed on specific machine pools.

4.4. Integration of the FMS Controller with a Production Planning System

Two problems arise in the integration of the proposed FMS controller with an MRP system (see Fig. 1). They will be referred to as the part loading problem and the part dispatching problem. Kimemia and Gershwin (1983) and Gershwin (1994) presented a control-theoretic approach for solving these two problems. Our approach is based on theirs, but we are interested in integrating the loading and dispatching rules with the FMS controller formulation presented in this paper. This integration actually closes the loop between the MRP system at a higher level in the manufacturing planning and control system and the controller used in the shop floor. We assume that the production quantities have been determined based on an algorithm that takes into account production capacities and demands, such as Gershwin's LP method [5].

4.4.1. Part loading

A part loading rule can be developed to attempt to minimize the difference between the scheduled order releases and the quantities actually released to the FMS cell. For this purpose define the discrete function of time

P_{ijt}^{IN} = final total number of parts type i started (loaded) in cell j during time period t, and define

$$T_{ijt}^{\mathrm{IN}} = [\sum_{k=1}^{t-1} (Z_{ijk} - P_{ijk}^{\mathrm{IN}}) + Z_{ijt}]^{+}$$

as the target number of parts type i that should be released to cell j up to period t. Our loading rule at any point of time during time period t is to load part i into cell j if

$$i = \arg \max\{[T_{ijt}^{\mathrm{IN}} - P_{ij}^{\mathrm{IN}}(\theta)]^{+}\}; \qquad \theta \in (t\text{-}1,t) \qquad (19)$$

where the variable $P_{ij}^{\mathrm{IN}}(\theta)$ is a continuous function of time and denotes the cumulative number of parts type i loaded in cell j at any point of time θ during period t. The relation between P_{ijt}^{IN} and $P_{ij}^{\mathrm{IN}}(\theta)$ is that at $\theta = t, 2t, 3t, \ldots$ we make $P_{ijt}^{\mathrm{IN}} = P_{ij}^{\mathrm{IN}}(\theta)$ and $P_{ij}^{\mathrm{IN}}(\theta) = 0$ (i.e., we restart at zero every t time units).

The time index t usually is measured in weeks and corresponds to the time units used by the MRP system.

4.4.2. Part dispatching

Once parts are loaded into the FMS cell, dispatching rules are invoked for conflict resolution. We can derive an additional dispatching rule based on information taken from the MRP system. In this case the rule will attempt to minimize the difference between the parts *produced* and the parts that were planned to be completed based on MRP information, as given by the scheduled order receipts. Define the discrete function of time

P_{ijt}^{OUT} = final total number of parts type i produced by cell j during period t, and define also

$$T_{ijt}^{\mathrm{OUT}} = [\sum_{k=1}^{t-1} (Q_{ijk} - P_{ijk}^{\mathrm{OUT}}) + Q_{ijt}]^{+}$$

as the target number of parts type i that should be produced in cell j up to period t. Then at any point of time $\theta \in (t\text{-}1,t)$ when there is a conflict within the cell; we can choose to give higher priority to part type i if

$$i = \arg\ \max\{[T_{ijt}^{OUT} - P_{ij}^{OUT}(\theta)]^+\}; \qquad \theta \in (t\text{-}1, t) \quad (20)$$

In this case, at = t, $2t$, $3t$, ..., we make

$$P_{ijt}^{OUT} = P_{ij}^{OUT}(\theta) \quad \text{and} \quad P_{ij}^{OUT}(\theta) = 0,$$

where $P_{ij}^{OUT}(\theta)$ is a continuous function of time and denotes the cumulative number of parts type i completed in cell j at any point of time θ during period t.

Obviously, this dispatching rule can be used in conjunction with some of the other dispatching rules presented before.

4.4.3. Part tracking

The quantitative variables $P_{ij}^{IN}(\theta)$ and $P_{ij}^{OUT}(\theta)$ can be related to the binary representation of the FMS controller, thus we now define an increment function as

$$\text{INC: } R^+ \times B \to N_0 = N \cup \{0\},$$

$$\text{INC}(x,b) = \begin{cases} x+1, & \text{if } b = \text{true} \\ x, & \text{if } b = \text{false} \end{cases} \quad (21\text{-}1)$$

$$\text{Thus,} \quad P_{ij}^{IN}(t) = \text{INC}(P_{ij}^{IN}(t), Pin_i)$$

$$\text{and} \quad P_{ij}^{OUT}(t) = \text{INC}(P_{ij}^{OUT}(t), Pout_i), \quad (21\text{-}2)$$

where Pin_i and $Pout_i$ are logical variables from the FMS controller description.

This permits the integration of upper production control systems (e.g. MRP) with the controller design structure. Such a function transfers the binary representation in the controller design to a numerical domain in order to monitor the number of parts loaded to the workcell as well as completed in a cell.

5. Conclusions

In this paper we first studied some basic manufacturing structures, providing matrix formulations for their description. Included were the Single-Machine-Single-Part-Type (SMSP), the Single-Machine-Multiple-Part-Type (SMMP), and the Multiple-Machine-Single-Part-Type (MMSP) structures. It was indicated how to combine these into the Multiple-Machine-Multiple-Part-Type (MMMP) system using an operation sequencing matrix S_u. Depending on the properties of S_u, the resulting structure was either a reentrant flow or a job-shop with alternate routings. The external decision inputs u_C and u_R are introduced respectively to deal with shared resources and route selection. These two problems were treated as dual problems thus dispatching rules can be used to solve both.

We also presented the integration of the FMS controller with an MRP system at a higher level in the manufacturing planning. This provides numerical computation in controlling part inputs and capacity analysis. Based upon the part loading and dispatching rules, parts were selected to match the scheduled demands.

6. References

[1] D.D. Bedworth, and J.E. Bailey, Integrated Production Control Systems: Management, Analysis, Design, 2/E, John Wiley & Sons, 1987.

[2] D.D. Cofer, and V.K. Garg, "A timed model for the control of discrete event systems involving decisions in the max/plus algebra," Proc. 31st Conf. on Decision and Control, pp. 3363-3368, 1992.

[3] E.A. Elsayed, and T.O. Boucher, Analysis and Control of Production Systems, 2nd edition, Englewood Cliffs, NJ: Prentice Hall, 1994.

[4] S.D. Eppinger, D.E. Whitney, R.P. Smith, and D.A. Gebala, "Organizing the tasks in complex design projects," Design Theory and Methodology-DTM '90, vol. 27, pp. 39-46, 1990.

[5] S.B. Gershwin, Manufacturing Systems Engineering, New Jersey: PTR Prentice Hall, 1994.

[6] H.-H. Huang, N. Barzin, F.L. Lewis, "A matrix framework for controller design for flow shop scheduling of FMS," Proc. ISPE/IFAC International Conf. on CARs & FOF'94, pp. 111-118, 1994.

[7] J. Kimemia and S.B. Gershwin, "An algorithm for the computer control of a flexible manufacturing system," IIE Transactions, vol. 15, no. 4, pp. 353-362, 1983.

[8] A. Kusiak, and J. Ahn, "A resource-constrained job-shop scheduling problem with general precedence constraints," working paper, no. 90-03, Iowa, 1991.

[9] C.-Y. Lee, and J.W. Herrmann, "Decision support systems for dynamic job-shop scheduling," Proc. NSF Design & Manufacturing Systems Conf., vol. 2, pp. 1119-1123, 1993.

[10] F.L. Lewis, H.-H. Huang, and S. Jagannathan, "A systems approach to discrete event controller design for manufacturing systems control," Proc. American Control Conf., pp. 1525-1531, 1993.

[11] S.S. Panwalkar, and W. Iskander, "A survey of scheduling rules," Operations Research, vol.25, no.1, 1977.

[12] O.C. Pastravanu, A. Gürel, H.-H. Huang, and F.L. Lewis, "Rule-based controller design algorithm for discrete event manufacturing systems," Proc. American Control Conf., pp. 299-305, 1994.

[13] O.C. Pastravanu, A. Gürel, and F.L. Lewis, "Petri net based deadlock analysis in flowshops with Kanban-type controllers," Proc. ISPE/IFAC International Conference on CARs & FOF '94, pp. 75-80, 1994.

[14] D.V. Steward, "The design structure system: a method for managing the design of complex systems," IEEE Trans. on Engineering Management, EM-28, no. 3, pp. 71-74, 1981.

[15] J.N. Warfield, "Binary matrices in system modeling," IEEE Trans. Systems, Man, and Cybernetics, SMC-3, no. 5, pp. 441-449, 1973.

[16] S.D. Wu, J. Leon, and R.H. Sturer, "Scheduling, control & rescheduling methodologies for job-shop in the presence & disruptions," Proc. NSF Design & Manufacturing Systems Conf., pp. 947-959, 1992.

[17] R.A. Wysk, N.S. Yang, and S. Joshi, "A procedure for deadlock detection in FMSs," IEEE transactions on Robotics and Automation, vol. 7, no. 6, pp. 853-859, 1991.