Design of Optimal Sequence Controller for A Flexible Manufacturing System

Huanxin Henry Xiong* MengChu Zhou* Reggie J. Caudill#
Center for Manufacturing Systems
*Department of Electrical & Computer Engineering
#Department of Industrial and Manufacturing Engineering
New Jersey Institute of Technology, Newark, NJ 07102-1982, USA

ABSTRACT *The optimal controller synthesis problem for discrete event control of a flexible manufacturing system (FMS) is presented in this paper. To explicitly formulate concurrent activities, multiple resources sharing, precedence constraints and dynamic routing in FMS operation, we adopt timed (place) Petri nets for problem representation. The A* based heuristic search algorithm is proposed to search for an optimal event sequence to achieve minimum-time and deadlock-free discrete event control. Based on the obtained event-driven sequence, we use two levels of specification to design the optimal sequence controller. The coordination control level consists of synchronization and parallelism of different sub-systems and is specified by decision-free Petri nets (marked graphs). The local control level consists of running elementary sequences for sub-systems and is specified by sequential function charts.*

1 Introduction

For discrete event control of FMS, an optimal control problem is to find an input event sequence that moves the system from a given initial state to a given final state while minimizing certain performance indexes. Various notions of optimal control have been studied for discrete event systems (DESs). Passino and Antsaklis (1989) used valid behavior model and allowable behavior to describe DES and proposed a metric space approach to heuristic search for an optimal solution. Lin and Ionescu (1992) considered optimization of controller design for discrete event systems in a temporal logic framework. Sengupta and Lafortune (1991) proposed graph-theoretic formulation of optimal discrete event control problems for a class of DESs.

Petri net theory has been applied for scheduling and discrete event control of flexible manufacturing system. Petri nets can concisely model the concurrent and asynchronous activities, resource sharing, and precedence constraints in FMS. Venkatesh, Zhou and Caudill (1994) identified certain criteria to compare ladder logic diagrams and Petri nets for sequence controller design through a discrete manufacturing system and proposed a real-time Petri nets for sequence control. Zhou and DiCesare (1992) presented a hybrid synthesis methodology to design a bounded, live and reversible Petri net controller. But for the system with multi-layer resource-sharing and different products sets manufactured concurrently, modeling of a Petri net with desirable properties becomes difficult based on this hybrid method. Hillion and Proth (1989) used timed event-graphs, a special class of timed Petri nets, for modeling and analyzing job-shop systems. Sayat and Ladet (1993) employed colored Petri nets and Grafcet to describe different levels of production control to deal with different levels of complexity presenting at each level. Lee and DiCesare (1994) presented a Petri net-based heuristic scheduling method for flexible manufacturing, although it does not guarantee to terminate with an optimal solution. Hybrid heuristic search was reported for FMS scheduling (Xiong et al., 1996).

This paper formulates and solves the optimal discrete event controller synthesis problem for a flexible manufacturing system in a timed Petri net framework. The bottom-up method is used to synthesize the system. Once the modeling is done, the A* based heuristic search algorithm which is combined with the execution of the timed Petri nets is proposed to search for an optimal event sequence to achieve minimum-time discrete event control. Based on the obtained event-driven sequence, we use two levels of specification to design the optimal sequence controller for the presented FMS.

Section 2 describes design method for discrete event control. A heuristic algorithm combined with the execution of the timed Petri nets for searching for an optimal event sequence is presented in Section 3. Section 4 illustrates a hierarchical method to specify an optimal controller for a flexible manufacturing system. Finally, Section 5 makes some conclusions.

2 Design Method For Discrete Event Control

A. Description of Design Procedure

Due to its complexity, the control of a flexible manufacturing system is commonly decomposed into a hierarchy of decision levels, such as planning, scheduling, supervisory control, and local control. This paper focuses on optimal sequence control problem in

0-7803-3280-6/96/$5.00 ©1996 IEEE

FMSs at the levels of scheduling, supervisory and local control. The optimal control problem is to find an input event sequence that moves the system from a given initial state to a final state while minimizing certain performance indexes. Based on the optimal event sequence, a sequence controller is designed for optimization of system performance. We use two levels of specification to design the optimal sequence controller. It is assumed that a host computer is responsible for coordination and synchronization of different sub-systems, such as machines, robots and AGVs. The control sequence implemented at this level is an optimal event sequence and can be specified by a decision-free Petri net (marked graph). The local control level consists of running elementary sequences for sub-systems. The sequence of operations executed by a local controller is specified by a sequential function chart (SFC) from which the controller program code such as relay ladder logic program, can be directly derived and implemented into a Programmable Logical Controller (PLC).

The design procedure for optimal sequence controllers is proposed as follows:

Step 1. Modeling of an FMS using timed Petri nets. The synthesis of Petri net models is based on a bottom-up approach which begins with the construction of subnets for component processes and proceed to the final net by merging and/or linking all these subnets. The concurrency, conflicts, resource-sharing, and sequential operations are concisely represented in the model.

Step 2. Heuristic search of the reachability graph of a timed Petri net model for an optimal or near-optimal event sequence. All feasible event sequences are incorporated in the reachability graph of the Petri net model resulted from *Step 1*. The search for an optimal event sequence is NP-complete, therefore, the heuristic search methods are employed.

Step 3. Synthesis of a choice-free Petri net model (marked graph) for event-driven coordination control based on the optimal event sequence. The event sequence obtained from *Step 2* optimally resolves the conflicts competing for shared resources among the processes. As a result, the system behavior can be described by a marked graph in which each place has exactly one input and one output transition. A marked graph is guaranteed to be live if and only if every circuit contains at least one token. This greatly reduces the analytical overhead for eliminating the deadlock states in the system. Therefore, compared with existing Petri net or other methods (Banaszak and Krogh 1990, Narahari and Viswanadham 1990, Zhou and DiCesare 1992), real-time control implementation of a marked graph can easily guarantee deadlock-free system behavior. Moreover, there exist effective methods for performance analysis of timed marked graphs (Hillion and Proth 1989).

Step 4. Specification of local sequence controllers for each sub-system using sequential function charts. SFC is a standard for describing the control logic of manufacturing devices. It overcomes two drawbacks inherent in Petri nets: nondeterministic evolution and infinite creation of tokens. In SFC, transition firing is synchronous, and a step can only be active or inactive.

B. Petri Net Modeling

For a given system, we construct its Petri net model based on the bottom-up method, i.e., a system is partitioned according to the job types, then a sub-model is constructed for each job type according to a general methodology (Zhou and DiCesare 1993), and finally a complete net model for the entire system is obtained by merging Petri nets of job types through the places representing the shared resources. When an FMS consists of many machines and deals with many types of jobs, modeling of a Petri net based on the above synthesis method cannot guarantee the liveness of the resultant model.

A firing sequence of the transitions from an initial one to a final marking can be obtained by searching over the reachability graph of the Petri net model if it exists. The sequence is then used to synthesize a decision-free and deadlock-free Petri net model for supervisory coordination control.

3 Heuristic Algorithm for Optimization of Event Sequence

For DES, an optimal control problem is to find an input event sequence that moves the system from a given initial state to a final state while optimizing a pre-defined performance index. Based on the obtained optimal sequence, a sequence controller can be designed.

An optimal event sequence is sought in a timed Petri net framework to achieve minimum-time control. In the Petri net model of a system, firing of an enabled transition changes the token distribution (marking). A sequence of firings results in a sequence of markings, and all possible behaviors of the system can be completely tracked by the reachability graph of a net. The search space for the optimal event sequence is the reachability graph of the net, and the problem is to find a firing sequence of the transitions in the Petri net model from the initial marking to the final one. A heuristic search algorithm is developed by combining the Petri net execution and a best-first graph search algorithm A*. The most important aspect of the algorithm is the elimination from further consideration of some subsets of markings which may exist in the entire reachability graph. Thus the amount of computation and the memory requirements is reduced.

Algorithm:

1. Put the start node (initial marking) m_0 on OPEN.
2. If OPEN is empty, exit with failure.

3. Remove from OPEN and place on CLOSED a marking m for which f is minimum.
4. If marking m is a goal node (final marking), exit successfully with the solution obtained by tracing back the pointers from marking m to marking m_0.
5. Otherwise find the enabled transitions of the marking m, generate the successor markings for each enabled transition, and attach to them pointers back to m.
6. For every successor marking m' of marking m:

(a) Calculate $f(m')$.

(b) If m' was neither on OPEN nor on CLOSED, add it to OPEN. Assign the newly computed $f(m')$ to marking m'.

(c) If m' already resided on OPEN or CLOSED, compare the newly computed $f(m')$ with the value previously assigned to m'. If the old value is lower, discard the newly generated marking. If the new value is lower, substitute it for the old and direct its pointer along the current path. If the matching marking m' resided on CLOSED, move it back to OPEN.

7. Go to step 2.

The function $f(m)$ in the above algorithm is a sum of two terms $g(m)$ and $h(m)$. $f(m)$ is an estimate cost (makespan) from the initial marking to final one along an optimal path which goes through the marking m. The first term, $g(m)$, is the cost of a firing sequence from the initial marking to current one. The second term, $h(m)$ is an estimate cost of a firing sequence from current marking m to the final marking, called *heuristic function*. The following heuristic function is used:

$$h(m) = \max_i \{ \xi_i(m),\ i = 1, 2, ..., N. \}$$

where $\xi_i(m)$ is the addition of operation time of those remaining operations for all jobs which are planned to be processed on the ith machine when the current system state is represented by the marking m. N is the total number of machines. The purpose of a heuristic function is to guide the search process in the most profitable direction by suggesting which transition to fire first.

For the above heuristic function, $h(m)$ is a lower bound to all complete solutions descending from the current marking, i.e.,

$$h(m) \leq h^*(m), \forall m$$

where $h^*(m)$ is the optimal cost of paths going from the current marking m to the final marking. Hence, the employed heuristic function $h(m)$ is admissible, which guarantees for an optimal solution (Pearl 1984).

The list OPEN maintains markings that have been generated and had the heuristic function applied to them. It chooses which marking to expand next based on the combination of how good the marking itself looks (as measured by $h(m)$) and how good the path to the marking was (as measured by $g(m)$). If the newly generated marking is already on OPEN, it means a new firing sequence (path) to this marking from initial marking has been found. The path is updated to yield the smallest cost whenever the new path has a cost lower than the old path.

The list CLOSED maintains markings that have already been examined. When a new marking is generated, it is checked whether the marking has been generated before. If the newly generated marking is on CLOSED and the new path has a cost lower than the old path, this marking is put in OPEN for re-exploration.

At each step of the best-first search process, the most promising of the markings generated so far is selected. The reachability graph grows from the initial marking until it touches the final one. Because of the heuristic function, only a portion of the reachability graph is generated. The more informed a heuristic function is, the smaller the number of generated markings is.

4 Illustration Through A Flexible Manufacturing System

The design procedure presented in Section 2 is illustrated through an FMS. The layout of a flexible manufacturing system is shown in Figure 1. It consists of two entries, two exits, three machines, three robots, and a two AGV system. Two job (product) types J_1 and J_2 are to be carried out. The precedence relationships among the operations and working time of each operation on the assigned machine for each job are shown in Table 1. Robot R_1 shared by M_1 and M_2 can be used to load M_1, to deliver raw material of product J_1 from Entry 1, and unload M_2 to send finished product J_2 fixtured to pallet to AGV 2. Robot R_2 is used to load M_3, to deliver raw material of product J_2 from Entry 2, and unload M_3 to send finished product J_1 fixtured to pallet to AGV 1. Robot R_3 is shared by M_1, M_2 and M_3 to convey intermediate parts. It performs the following functions: unloading M_1, loading M_2, unloading M_2, loading M_3 for job type J_1, and unloading M_3, loading M_1, unloading M_1, loading M_2 for job type J_2. Two AGVs have one pallet position each and are designed for the delivery of final parts and the release of pallets in the system. From M_3, AGV1 sends final product J_1 to the Exit 1 and pallet back to Entry 1. From M_2, AGV2 sends final product J_2 to Exit 2 and pallet back to Entry 2. Since they take different paths, collision is avoided and both AGVs can work concurrently.

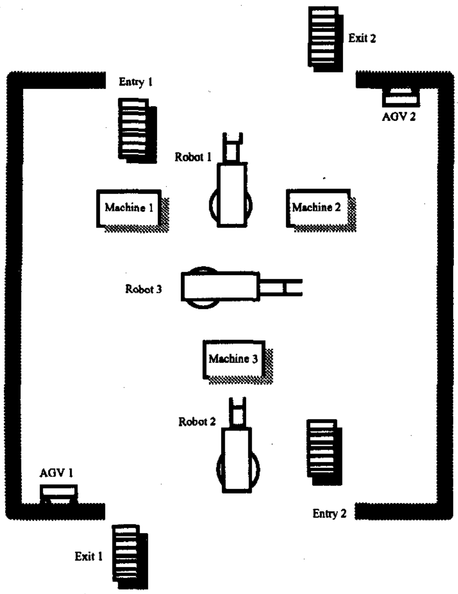


Figure 1 The layout of a flexible manufacturing system

Table 1. Job Requirements

Operation/Job	J_1	J_2
1	$(M_1,5)$	$(M_3,7)$
2	$(M_2,8)$	$(M_1,3)$
3	$(M_3,2)$	$(M_2,9)$

A. Petri Net Modeling

Based on the modeling method presented before, the Petri net models representing operation sequences for sub-system Job J_1 and J_2 are shown in Figure 2. The complete model for the entire automated manufacturing system is represented by merging the same places representing the shared resources in the Petri net models for sub-system Job J_1 and J_2.

B. Heuristic search based on timed Petri nets

The computation for optimal event sequence is conducted in *C* on a DEC 5900. Using the algorithm proposed in Section 3, we obtain the following optimal input event sequences for cyclic production:

Machine 1: <*Operation 1 of Job 1, Operation 2 of Job 2*>; Machine 2: <*Operation 2 of Job 1, Operation 3 of Job 2*>; Machine 3: <*Operation 1 of Job 2, Operation 3 of Job 1*>; Robot 1: <*Acquiring from Entry 1, Loading Machine 1, Unloading Machine 2, Loading AGV 2*>; Robot 2: <*Acquiring from Entry 2, Loading Machine 3, Unloading Machine 3, Loading AGV 1*>; Robot 3: <*Acquiring from Machine 1, Loading Machine 2, Acquiring from Machine 3, Loading Machine 1, Acquiring from Machine 2, Loading Machine 3, Acquiring from Machine 1, Loading Machine 2*>.

The concurrency of these events can be explicitly handled in the Petri net formalism. Based on the sequence control structure proposed in Section 2, two levels of specification, coordination control level and local control level are used to specify the optimal sequence controller.

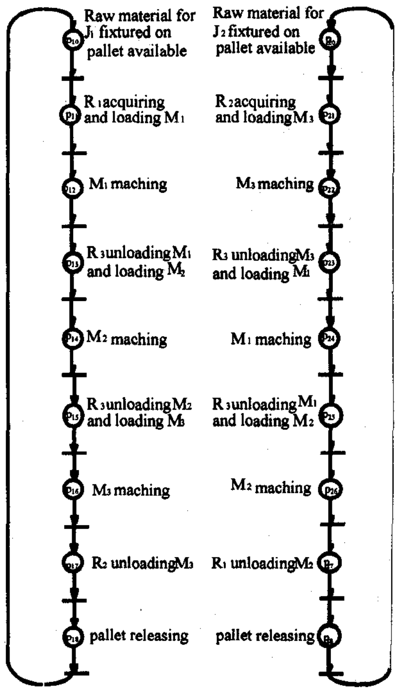


Figure 2 The operation sequences for Job 1 (*left*) and Job 2 (*right*)

C. Synthesis of Marked Graph for Event-Driven Coordination Control

Based on the obtained optimal event sequence, a choice-free Petri net model is synthesized for the case of both products having one pallet in the system. Figure 3 shows the Petri net model for coordination control which consists of synchronization and parallelism of different sub-systems. The presented Petri net is a marked graph in which each place has exactly one input and one output transition. The marked graph model of coordination control is developed as follows:

(i) Modeling the cyclic manufacturing process for each job type, we obtain the processing circuit $p_{10}t_{11}p_{11}t_{12}p_{12}t_{13}p_{13}t_{14}p_{14}t_{15}p_{15}t_{16}p_{16}t_{17}p_{17}t_{18}p_{10}$ for job type J_1, and the processing circuit $p_{20}t_{21}p_{21}t_{22}p_{22}t_{23}p_{23}t_{24}p_{24}t_{25}p_{25}t_{26}p_{26}t_{27}p_{27}t_{28}p_{20}$ for job type J_2. In each processing circuit, a place represents an event and a transition represents either start or completion of an event.

(ii) Modeling the sequencing of the part types for each machine according the optimal input event sequences above, three command circuits are obtained. The command circuit $c_{11}t_{11}p_{11}t_{12}p_{12}t_{13}c_{12}t_{23}p_{23}t_{24}p_{24}t_{25}c_{11}$ schedules the operations of Machine *1* and corresponding loading and unloading operations performed by robots. Similarly, the command circuit $c_{21}t_{13}p_{13}t_{14}p_{14}t_{15}c_{22}t_{25}p_{25}t_{26}p_{26}t_{27}c_{21}$ for Machine *2* and the command circuit $c_{31}t_{21}p_{21}$

$t_{22}p_{22}t_{23}c_{32}t_{15}p_{15}t_{16}p_{16}t_{17}c_{31}$ for Machine *3* are constructed.

(iii) Associating Boolean condition with transitions in the net, the logic condition of a transition can be all true logic *1* or the state of some specified steps of SFCs at the local control level.

D. Specification of Local Sequence Controllers

We use sequential function charts to specify local controllers. Figures 4(a) and (b) show the sequential function chart models for local control of Machines *1* and Robot *1* respectively. The detailed SFC models for all machines and robots are presented in [13]. The relation between two levels is realized by the logical conditions associated with some transitions in the coordination model and local control models. The Boolean variable *X(i)* is equal to *1* when and only when place (step) *i* is marked (active). For example, firing of transition t_{11} in Figure 3 marks place p_{11} and makes X(p_{11}) true. This initiates local controller of Robot *1* in Figure 4(b), which, in turn starts Robot 1 for picking up a part from Entry *1* and then loading Machine *1*. The event of end of loading Machine *1* makes step r_{14} active. This makes the condition related with transition t_{12} in coordination model true. Firing transition t_{12} marks place p_{12} and X(p_{12}) becomes true, which in turn makes Machine *1* process operation *1* of Job *1* based on the local controller of Machine 1 in Figure 4(a), and so on.

5 Conclusions

This paper investigates a design method for optimal and deadlock-free discrete event control of a flexible manufacturing system in a Petri net framework. Petri nets provide an efficient method for representing concurrent activities, shared resources, precedence constraints and routing flexibility in FMS. A heuristic algorithm is employed for searching optimal event sequences. A hierarchical method is presented to specify the optimal controller based on the obtained input event sequence. Considering different degrees of complexity at each level, we use different modeling tools to specify coordination level and local control level.

Further work will be conducted in developing more efficient heuristic algorithms for Petri net based optimization problems, and setting different performance indices such as minimum event cost control and due date control. The deadlock-free Petri net model of coordination control employing on-line sequencing rules will be developed. The performance of different coordination control strategies will be compared.

References

1] Banaszak, Z. and Krogh, B., "Deadlock Avoidance in Flexible Manufacturing Systems with Concurrently Competing Process Flows," *IEEE Trans. on Robotics and Automation,* 6(6), 1990, pp. 724-734.

[2] Hillion, H. P. and Proth, J. M., "Performance Evaluation of Job-Shop Systems Using Timed Event-Graphs," *IEEE Trans. on Automatic Control,* 34(1), 1989, pp. 3-9.

[3] Lee, D. Y. and DiCesare, F. "Scheduling FMS Using Petri Nets and Heuristic Search," *IEEE Trans. on Robotics and Automation,* 10(2), 1994, pp. 123-132.

[4] Lin, J. and Ionescu, D., "Optimization of Controller Design for Discrete Event Systems in A Temporal Logic Framework," *Proc. of the 1992 American Control Conference*, Chicago, Illinois, June 1992, pp. 2819-2823.

[5] Passino, K. M. and Antsaklis, P. J., "On the Optimal Control of Discrete Event Systems," *Proc. of the 28th Conf. on Decision and Control*, Tampa, Florida, Dec. 1989, pp. 2713-2718.

[6] Pear, J., Heuristics: *Intelligent Search Strategies for Computer Problem Solving*. Reading, MA, : Addison-wesley, 1984.

[7] Sayat, B. and Ladet, P., "Control Specification of A Production System Using Grafcet and Petri Nets," *RAIRO, Automatic Control Production Systems*, 27(1), 1993, pp. 53-64.

[8] Sengupta, R. and Lafortune, S., "Optimal Control of A Class of Discrete Event Systems," *IFAC Sym. on Distributed Intelligent Systems*, Arlington, Virginia, Aug. 1991, pp. 25-30.

[9] Venkatesh, K., Zhou, M. C. and Caudill, R. J., "Comparing Ladder Logic Diagrams and Petri Nets for Sequence Controller Design Through A Discrete Manufacturing Systems," *IEEE Trans. on Industrial Electronics*, 41(6), 1994, pp. 611-619.

[10] Viswanadham, N., Narahari, Y. and Johnson, T., "Deadlock Prevention and Deadlock Avoidance in Flexible Manufacturing Systems Using Petri Net Models," *IEEE Trans. on Robotics and Automation*, 6(6), 1990, pp. 713-723.

[11] Zhou, M. C., DiCesare, F. and Desrochers, A., "A Hybrid Methodology for Synthesis of Petri Nets for Manufacturing Systems," *IEEE Trans. on Robotics and Automation*, 8(3), 1992, pp. 350-361.

[12] Zhou, M. C. and DiCesare, F., *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems,* Kluwer Academic Publications, Boston, MA, 1993.

[13] Xiong, H., Zhou, M. C. and Caudill, R. J., "A Hybrid Heuristic Search Algorithm for Scheduling Flexible Manufacturing Systems." in IEEE Proc. of 1996 Int. Conf. on Robotics & Automation, Minneapolis, MN, April 22-28, 1996, Vol. 3, pp. 2793-2797.

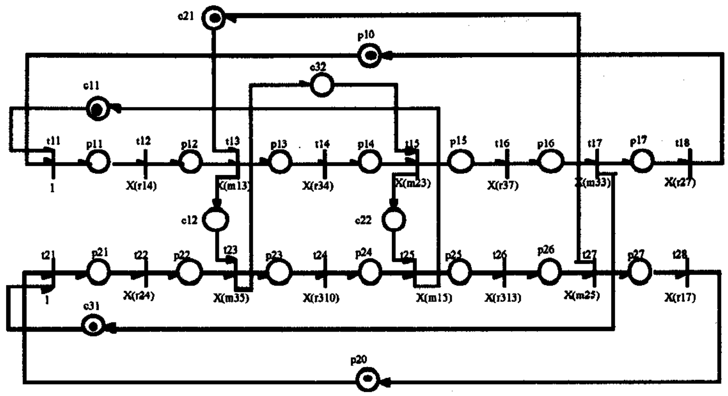
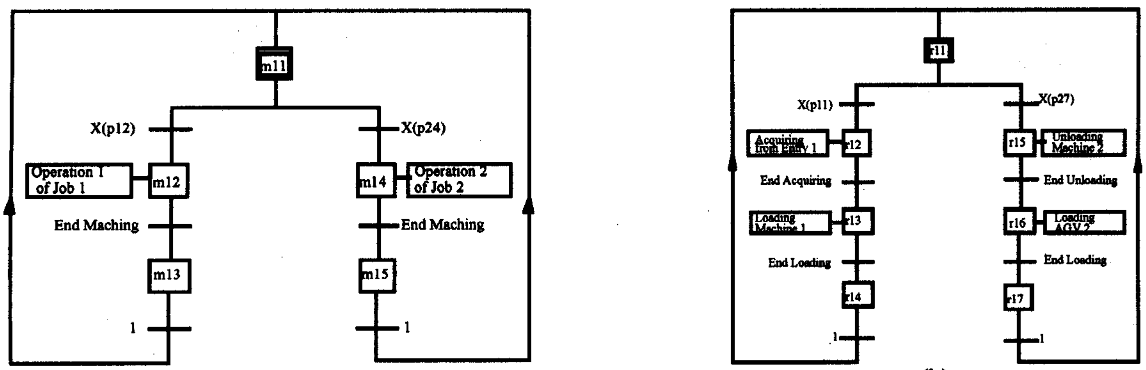


Figure 3 Petri net (Marked graph) model for coordination control



(a) (b)

Figure 4 The SFC models for local control of Machine 1(a) and Robot 1(b).