Computers & Industrial Engineering 71 (2014) 21–26

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Search strategy for scheduling flexible manufacturing systems simultaneously using admissible heuristic functions and nonadmissible heuristic functions ☆,☆☆

Bo Huang *, Rongxi Jiang, Gongxuan Zhang

School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, PR China

ARTICLE INFO

Article history:
Received 22 November 2013
Received in revised form 22 February 2014
Accepted 24 February 2014
Available online 3 March 2014

Keywords:
Petri net
Flexible manufacturing systems
A^* algorithm
Heuristic search

ABSTRACT

To scheduling flexible manufacturing system (FMS) efficiently, we propose and evaluate an improved search strategy and its application to FMS scheduling in the *P*-timed Petri net framework. On the execution of Petri net, the proposed method can simultaneously use admissible heuristic functions and nonadmissible heuristic functions for A^* algorithm. We also prove that the resulting combinational heuristic function is still admissible and more informed than any of its constituents. The experimental results of an example FMS and several sets of random generated problems show that the proposed search method performs better as we expected.

1. Introduction

Nowadays, flexible manufacturing system has been widely adopted in modern production environments to provide product variety and quick response to changes in marketplace. FMS is an automated manufacturing system where there may exist multiple concurrent flows of processes. Different products may be manufactured at the same time, and shared resources are often exploited to reduce the production cost. Therefore, the development of efficient planning and scheduling methods for FMS is an important issue.

In an FMS, there is a high-level scheduler that must decide what resources to be assigned to what job and at what time, so as that the makespan is minimized or the utilization of critical machines is maximized. But FMS scheduling problem belongs to one of the NP hard combinatorial problems (Tzafestas & Triantafyllakis, 1993), for which it is unlikely to develop an optimal polynomial algorithm.

To address the manufacturing system scheduling problem, Petri net (PN) has often been used (Dashora, Kumar, Tiwari, & Newman, 2007). PN (Murata, 1989) is a mathematical formalism and graphical tool that can be used for the modeling, design and analysis of discrete event systems. As a graphical tool, PN works like a flow chart to provide a visualization of a dynamic system. As a mathematical tool, PN model allows formal checking for properties of the behavior of the described system.

Based on the PN models for FMS, beam search, linear programming, dispatching rules and branch and bound methods have been studied by Shih and Sekiguchi (1991), Onaga, Silva, and Watanebe (1991), Camurri, Franchi, Gandolfo, and Zaccaria (1993) and Chen, Yu, and Zhang (1993) to find the optimized scheduling scheme. Although these methods use heuristic search in PN model, their performances were not good enough for FMS (Lee & Lee, 2010).

To find optimal or suboptimal scheduling sequences for FMS, Lee and Dicesare (1994) have combined PN simulation capabilities and A^* algorithm (Pearl, 1984) within the PN reachability graph. A^* is an informed search algorithm that uses a heuristic function to expands only the most promising branches of the PN reachability graph. Some admissible heuristic functions used in the A^* algorithm for FMS have been proposed by Xiong and Zhou (1998), Mejia (2002), Reyes, Yu, Kelleher, and Lloyd (2002), Yu, Reyes, Cang, and Lloyd (2003) and Lee and Lee (2010). With this kind of heuristic functions, A^* algorithm can guarantee that the solution found is optimal. However, a typical problem observed is that A^* algorithm with admissible function often costs much time to find

☆ This manuscript was processed by Area Editor Manoj Tiwari.

☆☆ This work was supported in part by National Natural Science Foundation of China (Grant Nos. 61203173, 61272420), National and Jiangsu Provincial Planned Projects for Postdoctoral Research Funds, Doctoral Fund of Ministry of Education of China (Grant No. 20103219120023) and the Zijin Intelligent Program of Nanjing University of Science and Technology.

* Corresponding author. Tel.: +86 25 84315660; fax: +86 25 84315636.
E-mail addresses: huangbo@njust.edu.cn (B. Huang), 811326603@qq.com (R. Jiang), gongxuan@njust.edu.cn (G. Zhang).

http://dx.doi.org/10.1016/j.cie.2014.02.010
0360-8352/© 2014 Elsevier Ltd. All rights reserved.

the optimal solution even for a medium size problem. To reduce the search time, Mejia (2002) and Lee and Lee (2010) also proposed some nonadmissible heuristic functions. These functions may invoke quicker termination conditions, but they cannot ensure the search results are optimal. In this paper, we propose a method for A^* algorithm to simultaneously use admissible heuristic functions and nonadmissible heuristic functions, and the combinational function is not only admissible but also more informed than its constituents.

This paper is organized as follows. FMS description and its PN model are presented in Section 2. In Section 3, a method of combine admissible heuristic functions and nonadmissible heuristic functions for A^* algorithm is presented and the properties of the combinational heuristic function are proved. In Section 4, several experimental results are shown and these results are compared with those of A^* algorithms in the literature. In Section 5, conclusions and future works are discussed.

2. FMS description and its PN model

In this section, we first describe the FMS we attempt to solve in this paper and derive its *P*-timed PN. Based on the PN model, we use A^* search algorithm to solve FMS scheduling problems.

A general FMS can be represented as below.

1. $R = \{R_1, R_2, \ldots, R_m\}$ is a set of m resources;
2. $J = \{J_1, J_2, \ldots, J_n\}$ is a set of n jobs describing part types;
3. Each job J_i has s_i sequences with lot size l_i. The lot size l_i means the number of part to be processed in the job J_i;
4. Each sequence S_{ij} has t_{ij} tasks ordered by the product processing procedures;
5. Each task T_{ijk} can be achieved in o_{ijk} number of ways. Each way means an operation that completes the task.
6. Each operation O_{ijkl} needs r_{ijkl} number of different resources to complete the operation and has a processing time p_{ijkl}.

Several reasonable assumptions are made as Reyes et al. (2002) for the above FMS description.

The FMS assumptions are as follows:

- Each machine (regarded as a resource) can process at most one task at a time and no pre-emption is allowed.
- Each task consumes a single subpart of the previous unit and produces only a single subpart (there is no assembling).
- An infinite buffer policy applies in the system. But different storage models can be used.
- Machine tool loading and set-up are considered negligible.

To solve the FMS scheduling, *P*-timed PN model and *T*-timed PN model are widely used in prior works. Since it is proved by Murata, 1989 that *P*-timed PN and *T*-timed PN are equivalent, we adopt *P*-timed PN which we have used in Huang, Sun, and Sun (2008), Huang, Sun, Sun, and Zhao (2010) and Huang, Shi, and Xu (2012) for convenience.

The definition of the general *P*-timed PN is presented as:

Definition 1. A general *P*-timed PN is a six-tuple $PPN = (P, T, I, O, M, d)$ where:

- $P = \{P_1, P_2, \ldots, P_m\}$ is a finite set of places;
- $T = \{T_1, T_2, \ldots, T_n\}$ is a finite set of transitions with $P \cup T \neq \varnothing$ and $P \cap T = \varnothing$;
- I: $P \times T \rightarrow \{0,1,2,\ldots\}$ is an input function or direct arcs from P to T;
- O: $T \times P \rightarrow \{0,1,2,\ldots\}$ is an output function or direct arcs from T to P;
- M: $P \rightarrow \{0,1,2,\ldots\}$ is a marking that indicates the number of tokens in each place. M_0 is the initial marking and M_G is the goal marking;
- d: $P \rightarrow R^{+} \cup \{0\}$ is a delaying function that associates the time delay with some places. Note that R^{+} is a set of positive real numbers.

In a *P*-timed PN for FMS, a place represents a resource status or an operation, a transition represents either start or completion of an event or operation process, and the stop transition for one activity will be the same as the start transition for the next activity following. Token(s) in a resource place indicates that the resource is available and no token indicates that it is not available. A token in an operation place represents that the operation is being executed and no token shows no operation is being performed. A certain time may elapse between the start and the end of an operation. This is represented by associating time delay with the corresponding operation place.

An illustrative example based on this model is shown below. Table 1 shows the requirements of each job in the example FMS. The FMS consists of three types of resource R_1, R_2, R_3 and four types of job J_1, J_2, J_3, J_4 with lot size 2, 2, 1, 1 respectively. Each job has only one sequence and each sequence has three tasks. Each task has no more than two operations and each operation uses one or two resources at a time. The processing time of each operation is represented by the number in parentheses. The modeling is briefed as follows. First, model a *P*-timed PN for each job based on their sequence and the use of resources. Then merge these models to obtain a complete model through the shared resource places which model the availability of resources. Fig. 1 shows the *P*-timed PN model for this system, where the resource places with a same name represent the same resource place. The intermediate buffers are represented by places P_{i3} and P_{i5} for $i = 1, 2, 3,$ and 4. No timing is associated with the intermediate buffers.

In scheduling such an FMS, we adopt the widely used *L*1 algorithm (Lee & DiCesare, 1994) which is an application of the well-known A^* algorithm to the FMS scheduling problem based on PN. The algorithm is as below.

(1) Put the initial marking M_0 on the list OPEN.
(2) If OPEN is empty, terminate with failure.
(3) Remove the first marking M from OPEN and put M on the list CLOSED.
(4) If M is the goal marking M_G, construct the scheduling path from M_0 to M_G and terminate.
(5) Otherwise, expand M: Find the enabled transitions of the marking M; Generate the next marking M', or successor, for each enabled transition, and set pointers from the next markings to M; Compute $g(M')$ for every successor M'. Note that $g(M)$ is the actual cost generated while transferring from the initial marking M_0 to the current marking M.

Table 1
A simple FMS example.

Tasks/jobs	J_1 Sequence1	J_2 Sequence1	J_3 Sequence1	J_4 Sequence1
Task 1	$R_1(6)$ or $R_2(7)$	$R_2(4)$	R_1 and $R_2(3)$	$R_3(3)$
Task 2	R_2 and $R_3(5)$	$R_2(3)$ or $R_3(4)$	$R_2(5)$	$R_3(4)$
Task 3	R_1 and $R_2(4)$ or $R_3(2)$	$R_1(2)$ or $R_3(4)$	$R_1(4)$ or R_2 and $R_3(2)$	$R_1(6)$