Multiobjective Heuristic Scheduling of Automated Manufacturing Systems Based on Petri Nets

Chong Yu, Bo Huang, and Jiangen Hao

***Abstract*—In practice, automated manufacturing systems usually have multiple, incommensurate, and conflicting objectives to achieve. To deal with them, this paper proposes an extend Petri nets for the multiobjective scheduling of AMSs. In addition, a multiobjective heuristic A* search within reachability graphs of extended Petri nets is also proposed to schedule these nets. The method can obtain all Pareto-optimal schedules for the underlying systems if admissible heuristic functions are used. Finally, the effectiveness of the method is illustrated by some experimental systems.**

***Index Terms*—Automated manufacturing systems, multiobjective heuristic search, Pareto-optimal schedules, Petri nets.**

I. Introduction

Automated Manufacturing Systems (AMSs) are a kind of computer-controlled systems that consist of limited resources and can handle different types of parts. In order to execute the automated manufacturing system effectively and make full use of system resources, it is necessary to coordinate and control the use of shared resources. However, this scheduling problem is NP-hard, because the computational time increases exponentially with system size [1].

Petri nets (PNs) are a graphical and mathematical modeling tool that is suitable for modeling distributed, concurrent, parallel, asynchronous in discrete event systems. Recently, they have become a popular tool to model and analyze AMSs [2]-[5]. Petri nets can concisely describe the activities, resources, and constraints in such systems.

Based on Petri nets, Lee *et al*. [6] propose a scheduling method that execute an intelligent search within the reachability graphs to schedule AMSs. It uses A∗ search and heuristic functions to restrict the search space. Base on the method, some improve methods are proposed in literature [7]-[11]. We also developed some approaches to improve the search process, such as a hybrid heuristic A* search [12], dynamic weighted A* search [13], and more informed heuristics [14].

However, the above methods focus on single-objective scheduling problem for AMSs. In practice, the scheduling of AMSs often includes multiple, incommensurate, and, conflicting objectives, such as cost, makespan, and tardiness. When compared with the single-objective approaches, the multiobjective scheduling problems are more difficult and often need to search and find a set of Pareto-optimal or nondominated schedules. In this paper, we propose a multiobjective A* search algorithm within reachability graphs of Petri nets to obtain Pareto-optimal schedules for AMSs.

II. Preliminaries

A. Petri Nets

A Petri net [3] is defined as a four-tuple $N = (P, T, F, W)$ where $P = \{p_1, p_2, \cdots, p_m\}, m > 0$ is a set of places; $T = \{t_1, t_2, \cdots, t_n\}$, $n > 0$ is a set of transitions such that $P \cap T = \emptyset$; $F \subseteq (P \times T) \cup (T \times P)$ is the set of directed arcs connecting places and transitions; W: $(P \times T) \cup (T \times P) \rightarrow N$ is a weight assignment for all arcs. For a net N, $M : P \rightarrow N = N+\cup \{0\}$ is called a marking that is a token distribution in P. M_0 is called an initial marking of N. At a marking M, if $\forall p \in \bullet t, M(p) > W(p, t)$, we say that t is enabled at M, denoted as $M[t\rangle$. If an enabled transition t fires at M, it generates a new marking M^* such that $\forall p \in P, M^*(p) = M(p) - W(p, t) + W(t, p)$. In this case, M^* is said to be reachable from M. For a net N with M_0, $G(N, M_0)$ called a reachability graph of the net in which each vertex represents a marking reachable from M_0 and each edge denotes a marking transfer by firing a transition.

In literature, several classes of PNs have been proposed for the control of AMSs, such as S^3PR [15], S^4PR [16], and S^*PR [17]. All of them consist of several state machines that share a set of resources. Their places are divided into three disjointed types: idle places P_0, operation or activity places P_A, and resource places P_R. The differences between them are mainly the number and types of resources that can be used at each processing step of a part and the structures of processing subnets.

B. Heuristic A* Search

There are several informed heuristic graph search algorithm, such as best-first (BF), BF*, Z, Z*, and A*, and they use heuristic information to decide which node to expand next.

Manuscript received July 18, 2019; revised September 22, 2020. This work was supported in part by National Natural Science Foundation of China under Grant 61773206, Natural Science Foundation of Jiangsu Province under Grant BK20170131, and Foundation of Fujian Engineering Research Center of Motor Control and System Optimal Schedule (Huaqiao University) under Grant FERC002.

C. Yu and B. Huang are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (Corresponding author: B. Huang; e-mail: yuchong@njust.edu.cn; huangbo@njust.edu.cn).

J. G. Hao is with the Nanjing Les Information Technology Co. Ltd., Nanjing 210007, China (e-mail: hjgnj@gmail.com).

Among them, A* is the most popular heuristic search algorithm in use.

By using an admissible heuristic function, the A* algorithm only needs to explore a partial graph to find an optimal schedule from an initial node to a goal node if such a schedule exists. It's used evaluation function is applied on each node *n*,

$$f(n) = g(n) + h(n)$$

where $g(n)$ is the current lowest cost obtained from the initial node to the current node *n*, $h(n)$ is a heuristic function that estimates of the lowest cost path from n to the goal node among all paths, and $f(n)$ is an estimate of the lowest cost from the start node to the goal node among all paths going through *n*. A* iteratively expands the search space from the start node until the node to be expanded reaches the goal node. Once the goal node is found, a path is constructed by tracing the pointers that denote the parenthood of the nodes, from the goal node to the start one. Then, the order of the executable activities, i.e., the system schedule, is obtained. In addition, the obtained schedule is optimal if the used heuristic function f is admissible, that is, for any reachable state *n*, $h(n)$ less than or equal to $h^*(n)$ in which $h^*(n)$ denotes the cheapest cost from *n* to the goal node.

Although Petri nets showed promise as an effective tool to formulate and solve the scheduling problems of AMS's, the actual generation of given much attention in the Petri net community. Recently, there have been some independent efforts to use Petri nets to generate schedules for AMS's.

Shih and Sekiguchi present an AMS scheduling system which simulates the evolution of the AMS as modeled by Petri nets. The scheduling system calls for a beam search routine whenever there is a conflict. The beam search routine then constructs partial schedules within the beam-depth and evaluates them to choose the best one. The cycle is repeated until a complete schedule is achieved. This method based on partial schedules does not guarantee global optimization. Onaga presents a linear programming approach for periodic scheduling of systems modeled by Petri nets. Shen presents a scheme which starts with an arbitrary schedule and applies branch and bound search to find an optimal schedule. Zhang presents a method which translates rules of a rule-based planning system into a timed Predicate/Transition net model and applies the A∗ algorithm to find an optimal schedule. The scheduling method presented in this paper formulates a scheduling problem using a Petri net model, and employs global search and limits the search space by the use of heuristic functions. The methods generate an optimal or near optimal feasible schedule in terms of the firing sequence of the transitions of the Petri net model. This method is also event driven as opposed to time driven, i.e., the schedule is provided as an order of the initiations of the activities. Most of the current scheduling approaches can be considered as time driven, i.e., the schedule is a list of time instants when certain activities are to happen. This approach may not always be best for the scheduling of automated manufacturing systems that are, by nature, discrete event driven. Event driven scheduling focuses on the precedence constraints of the activities and is robust to disturbances.

There are many targets for optimization in manufacturing. For example, the minimization of makespan and/or tardiness is one of the frequently adapted goals. The maximum utilization of critical machines is also often considered. Generating a schedule with the minimum or near minimum makespan is the focus in this paper.

In timed Petri nets, time can be associated with either places (timed place Petri nets), or transitions (timed transition Petri nets). Generally, a timed transition removes tokens from its input places and takes some time before it introduces tokens to its output places. Therefore, between the initiation and the termination of firing, the marking (the state of the system) is uncertain. Depending on whether timed transitions or timed places are used, activities are associated with transitions or places. respectively. In the case of timed transitions, multiple initiation or firing of transitions must be allowed to represent concurrency of activities. Therefore, the time associated with each initiated transition must be tracked in order to correctly update the marking, or state. Since the initiated transitions may not be tracked in applications of Petri net modeling, an additional tracking method is required.

III. Multiobjective A* Search with Extended Petri Nets

A. Extended Petri Nets

TABLE I: Attribute Matrix of the Example System

Place	Attributes	Place	Attributes	Place	Attributes
p1	(0, 0)	p8	(0, 0)	p15	(0, 0)
p2	(3, 1)	p9	(2, 1)	p16	(0, 0)
p3	(2, 7)	p10	(4, 3)	p17	(0, 0)
p4	(4, 1)	p11	(4, 2)	p18	(0, 0)
p5	(4, 3)	p12	(3, 3)	p19	(0, 0)
p6	(3, 2)	p13	(5, 1)	p20	(0, 0)
p7	(5, 4)	p14	(0, 0)	p21	(0, 0)

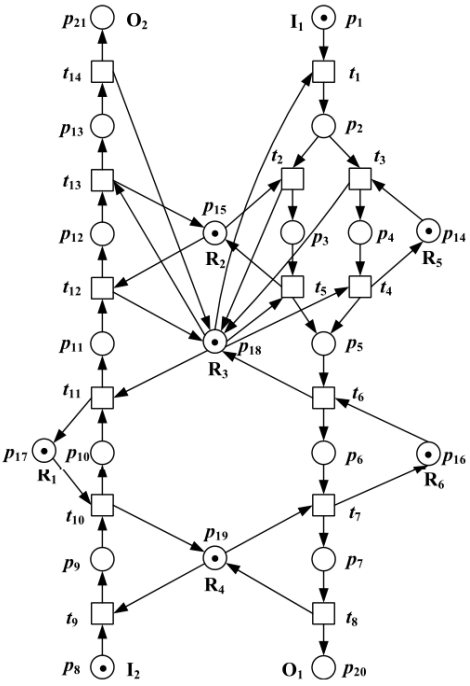

Fig. 1. Petri net model of the example system.

Since several objectives are considered in the AMS scheduling based on Petri nets, the classes of Petri nets used in

the literature are not suitable for the multiobjective search. Let u be the number of objectives to be considered in the scheduling problem. Let P_S be the set of start places that represent the start of jobs in an AMS and P_E be the end of start places that represent the end of jobs in the system. We define an extended *PN* for the multiobjective scheduling as (N, D) where $N = (P, T, F, W)$ with $P = P_S \cup P_E \cup P_A \cup P_R$ and D denotes a $|P| \times u$ attribute matrix on activity places, in which each row represents different non-commensurate costs on a place.

For example, consider an AMS adapted from [18] as an example. The AMS system has two robots R_3 and R_4, each of which holds one part at a time, and four machines R_1, R_2, R_5, and R_6, each of which processes one part at a time, two loading buffers I_1 and I_2, and two unloading buffers O_1 and O_2. Two part types, P_1 and P_2, are processed in the system by using the following routings. P_1 is taken from I_1 by R_3, and after being handled by R_2(or R_5) and R_6, it is moved to O_1 by R_4. P_2 is taken from I_2 by R_4, and after being handled by R_1 and R_2, it is moved to O_2 by R_3. For each part type, six part units are to be processed. Suppose that two objectives that minimize the cost of time and money are considered in the system scheduling. Its attribute matrix D for places is given in Table I where two elements in parentheses represent the operation time and money required by an operation depicted by a place. Its extended Petri net is shown in Fig. 1 where $P_S = \{p_1, p_8\}$, $P_E = \{p_{20}, p_{21}\}$, $PR = \{p_{14} - p_{19}\}$, and $PA = \{p_2 - p_7, p_9 - p_{13}\}$.

B. Domination Relations between Different Paths

Consider a finite set of objectives O_i, $i \in \{1,...,m\}$ with $m \in Z+$. A solution path can be evaluated by a cost vector $(C_1, ..., C_m) \in Z^m_+$ where C_i represents the cost of the path with respect to O_i. Thus, the comparison of different paths becomes the comparison of their cost vectors.

Definition 1: Let y, y^* be two different vectors in Z^m_+. We say y dominate y^*, denoted as $y \leqslant_p y^*$, if y is at least as good as y^* with respect to all elements, i.e.,

$$y \leqslant_p y^* \Leftrightarrow [\forall_i \in \{1,...,m\}, y_i \leqslant y^*_i].$$

Definition 2: Let Y be a set of vectors in Z^m_+ and y^* be a vector in Y. y* is said to be non-dominated (or Pareto-optimal) in Y if there does not exist another vector $y \in Y$ such that y dominates y^*.

Note that, in the reachability graph of an extended Petri net, there often exist different paths from a node to another node with different cost vectors. Accordingly, a path P in the reachability graph is said to be non-dominated (or Pareto-optimal) if there does not exist another path whose cost vector dominates the cost vector of P.

For any dominance relation defined on a set X, we will use the following definitions:

Definition 3: Any element $x \in X$ is said to be $\leqslant$-optimal in X if, for all $y \in X$, $y \leqslant x \Rightarrow x \leqslant y$. If x is not $\leqslant$-optimal then it is said to be $\leqslant$-dominated.

Definition 4: A subset $Y \subseteq X$ is said to be a $\leqslant$-covering of X if for all $x \in X$ there exists $y \in Y$ such that $y \leqslant x$. Whenever no proper subset of Y is a $\leqslant$-covering of X, then Y is said to be a minimal $\leqslant$-covering of X.

The aim of multiobjective search is to find a $\leqslant$p-covering of the set of solution-paths.

C. Multiobjective Scheduling Algorithm

In this subsection, a multiobjective A$*$ search (denoted as MOA$*$) within reachability graphs of extended Petri nets is proposed. The proposed algorithm can find all non-dominated paths from an initial state to a goal state in the reachability graph of a given extended Petri net for an AMS. The MOA$*$ algorithm is given in Algorithm I in which a state of the extended Petri net is defined as $S = (M, R)$ where M denotes a marking and R denotes remaining attribute matrix which represents remaining costs of each place at a specific marking.

Algorithm I: MOA$*$ Algorithm for Amss Based On Extended Petri Nets

Input: An extended Petri net of an AMS, an attribute matrix D, an initial state S_0, and a goal state S_G.

Output: All Pareto-optimal paths from S_0 to S_G.

1) Construct three empty lists: OPEN, CLOSED, and SOLUTION. Place the initial state S_0 on OPEN.

2) Remove the states in OPEN that is dominated by the states in SOLUTION.

3) If OPEN is empty, trace through back pointers from the state S in SOLUTION to the initial state S_0 and terminate the algorithm.

4) Remove the first state $S = (M, R)$ from OPEN and put it on CLOSED.

5) If $S = S_G$, add S to SOLUTION, remove the states that are dominated by S from SOLUTION, and go to Step 7).

6) For each transition t enabled at S: Generate a successor state S^* by firing t at S; Calculate $G(S)$, $H(S)$, $F(S) = G(S) + H(S)$ of S^*; Set a pointer from S^* to S.

a) If there exists a state S^{**} in OPEN such that $S^*.M = S^{**}.M$, compare their R and G values. If $S^*.R \leqslant_p S^{**}.R$ and $G(S^*) \leqslant_p G(S^{**})$, delete S^{**} from OPEN and insert S^* into OPEN; Otherwise, insert S^* into OPEN.

b) If there exist the state S^{**} in CLOSED such that $S^*.M = S^{**}.M$, compare their R and G values. If $S^*.R \leqslant_p S^{**}.R$ and $G(S^*) \leqslant_p G(S^{**})$, delete S^{**} from CLOSED and insert S^* to OPEN. Otherwise, insert S^* into OPEN.

c) If both OPEN and CLOSED do not contain a state S^{**} such that $S^*.M = S^{**}.M$, insert S^* into OPEN.

7) Reorder the states in OPEN in the following manner: First, reorder the states in the non-decreasing magnitude of the first attribute in F; If more than one state has the same value of the attribute, then rank them in the non-decreasing magnitude of the second attribute in F; and so on.

8) Go to Step 2).

In Algorithm I, the list OPEN contains nodes that are considered as potential candidates for expansion, the list CLOSED used in MOA$*$ to keep track of nodes that have been generated, but are not on OPEN, and the list SOLUTION are collection of solution paths that have been discovered prior to the start. Note that a transition t in an extended Petri net is enabled at a state $S = (M, R)$ if and only if $S.M[t\rangle$ and the remaining costs of the tokens required by the

firing of t are all zero. Introducing SOLUTION helps to collect multiple solutions and illustrates that there may be more than one nondominated paths with different attributes.

D. Heuristic Functions for Multiobjective Search

Similarly as the multiobjective A∗ algorithm in [19], the proposed MOA∗ search within the reachability graphs of extended Petri nets has an important property of Pareto-optimality.

Definition 5: In MOA∗, a heuristic function H is said to be admissible if $\forall S \in R(N, S_0)$, $H(S)$ is non-dominated in the cost vectors of all paths from the current state S to the goal state S_G.

Theorem 1: MOA∗ with an admissible heuristic function H can find all Pareto-optimal solutions if such solutions exist.

The admissible scheduling algorithm can always find an optimal path if $H(S)$ satisfies the following condition:

$$H(S) \leqslant_p H^*(S), \forall S$$

where $H*(S)$ is the attributes of optimal paths going from the current state S to the goal state SG. They aim to achieve a fairly good solution at a reasonable cost [20].

$$H(S) = (0, 0, ..., 0)$$

In the sequel, the heuristic function suitable for the MOA∗ are given.

IV. Experiments

This section tests some AMS examples to show the effectiveness of the proposed approach. First, we consider the aforementioned example system. Eight sets of lot sizes are tested by the MOA∗ algorithms with H. The cost vectors of Pareto-optimal schedules, the number of expanded states (N_S), and the computational time (T) are shown in Table II.

TABLE II: Multiobjective Scheduling Results for the Model in Fig. 1

p1	p8	Pareto results	N_S	T
1	1	(21, 17), (22, 15), (23, 11)	7.40×10^1	0.09s
2	2	(35, 23), (38, 20), (39, 18)	1.37×10^3	0.38s
3	3	(51, 30), (54, 27), (55, 25)	6.76×10^3	5.55s
4	4	(67, 37), (68, 36), (70, 34), (71, 32)	1.84×10^4	51.70s
5	5	(83, 44), (84, 43), (86, 41), (87, 39)	3.70×10^4	268.08s
6	6	(99, 51), (100, 50), (102, 48), (103, 46)	6.29×10^4	889.03s
7	7	(115, 58), (116, 57), (118, 55), (119, 53)	9.65×10^4	2269.90s
8	8	(131, 65), (132, 64), (134, 62), (135, 60)	1.38×10^5	5718.31s

The second AMS comes from [7] and it has an intermediate buffer between any two consecutive operations to hold parts that are ready for the next operation. The system consists of four input buffers I_1–I_4, four output buffers O_1–O_4, and three resources R_1–R_3. Four types of parts, J_1–J_4, are considered in the system. Their processing sequences are as below.

$$J_1: I_1 \rightarrow R_1 \rightarrow R_2 \rightarrow R_3 \rightarrow O_1$$

$$J_2: I_2 \rightarrow R_2 \rightarrow R_3 \rightarrow R_1 \rightarrow O_2$$

$$J_3: I_3 \rightarrow R_3 \rightarrow R_1 \rightarrow R_2 \rightarrow O_3$$

$$J_4: I_4 \rightarrow R_3 \rightarrow R_2 \rightarrow R_1 \rightarrow O_4$$

The attribute matrix is given in Table III. Note that the places with nonzero attribute vectors represent the operations to be performed with some specific resources. Its Petri net is shown in Fig. 2 which has 24 transitions and 31 places where $P_S = \{p_1, p_8, p_{15}, p_{22}\}$, $P_E = \{p_7, p_{14}, p_{21}, p_{28}\}$, $P_R = \{p_{29}, p_{30}, p_{31}\}$, and the rest places belong to activity ones P_A. For such a net, four sets of lot size are tested by the proposed MOA∗ algorithm with different heuristic functions. The multiobjective scheduling results are given in Table IV.

TABLE III: Attribute Matrix of the Net in Fig. 2

Place	Attributes	Place	Attributes	Place	Attributes
p1	(0, 0)	p12	(0, 0)	p22	(0, 0)
p2	(3, 2)	p13	(5, 4)	p23	(7, 1)
p3	(0, 0)	p14	(0, 0)	p24	(0, 0)
p4	(2, 3)	p15	(0, 0)	p25	(2, 1)
p5	(0, 0)	p16	(2, 2)	p26	(0, 0)
p6	(6, 4)	p17	(0, 0)	p27	(4, 2)
p7	(0, 0)	p18	(2, 2)	p28	(0, 0)
p8	(0, 0)	p19	(0, 0)	p29	(0, 0)
p9	(2, 1)	p20	(4, 5)	p30	(0, 0)
p10	(0, 0)	p21	(0, 0)	p31	(0, 0)
p11	(3, 1)				

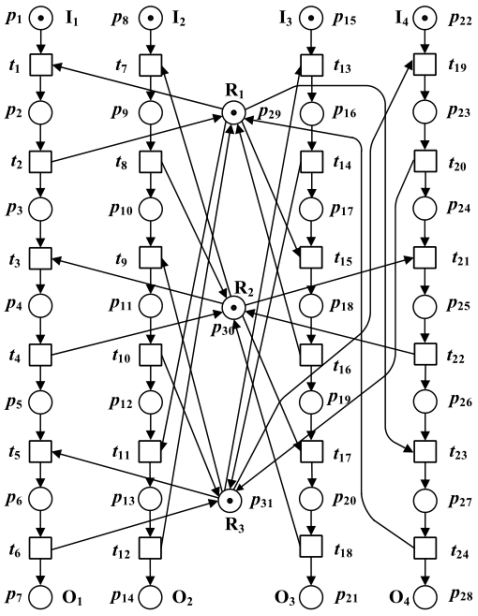

Fig. 2. A Petri net of an AMS [7].

The third AMS example is a more complex system adapted from [15]. The attribute matrix is given in Table V. Its Petri net model is shown in Fig. 3 where idle places in its original net have been split into start places and end ones for the sake of scheduling. The net has 20 transitions and 29 places in which start places are $\{p_1, p_5, p_{14}\}$, end places are $\{p_{27}–p_{29}\}$, resource places are $\{p_{20}–p_{26}\}$, and the rest places are activity ones.

Table VI to Table VII give the Pareto schedules in the form of a sequence of transition firings and its fire attributes when the lot size is fixed at 1 for each job. It has two Pareto results, which are (25, 17) and (27, 14), both of them can be the best.

TABLE IV: MULTIOBJECTIVE SCHEDULING RESULTS FOR THE NET IN FIG. 2

p1	p8	p15	p22	Cost vectors of Pareto-optimal schedules	N_S	T
1	1	1	1	(26, 22), (29, 20)	6.21×10^2	0.11s
2	2	2	2	(45, 34), (49, 33)	1.14×10^4	14.15s
3	3	3	3	(64, 52), (65, 51), (66, 50), (67,48)	6.73×10^4	1465.80s
4	4	4	4	(83, 67), (86, 63)	2.47×10^5	14363.24s

TABLE V: ATTRIBUTE MATRIX OF THE NET IN FIG. 3

Place	Attributes	Place	Attributes	Place	Attributes
p1	(0, 0)	p11	(1, 2)	p21	(0, 0)
p2	(2, 3)	p12	(3, 2)	p22	(0, 0)
p3	(4, 2)	p13	(4, 5)	p23	(0, 0)
p4	(5, 1)	p14	(0, 0)	p24	(0, 0)
p5	(0, 0)	p15	(6, 3)	p25	(0, 0)
p6	(3, 3)	p16	(3, 4)	p26	(0, 0)
p7	(2, 4)	p17	(4, 1)	p27	(0, 0)
p8	(6, 1)	p18	(6, 2)	p28	(0, 0)
p9	(2, 1)	p19	(2, 3)	p29	(0, 0)
p10	(4, 4)	p20	(0, 0)		

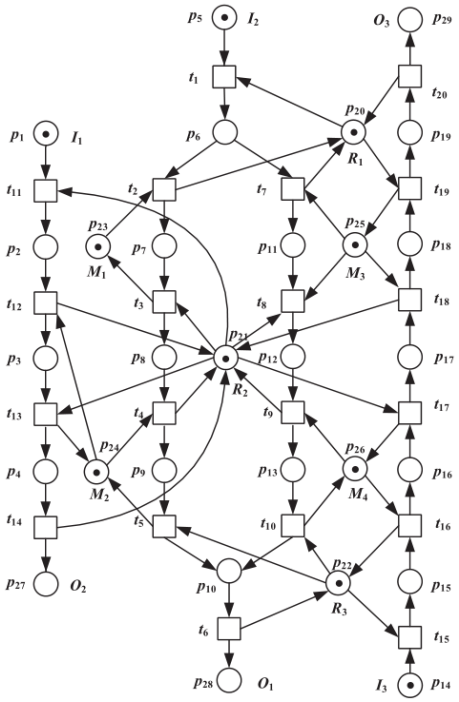

Fig. 3. A Petri net model of an AMS [15].

TABLE VI: A PARETO-OPTIMAL SCHEDULE FOR FIG. 3 (SCHEDULE RESULT = (25, 17))

Fire transition	Costs point	Fire transition	Costs point	Fire transition	Costs point
t15	(0, 0)	t11	(11, 8)	t13	(17, 13)
t1	(0, 0)	t5	(13, 9)	t6	(17, 13)
t2	(3, 3)	t12	(13, 11)	t14	(22, 14)
t3	(5, 7)	t17	(13, 11)	t19	(23, 14)
t16	(6, 7)	t18	(17, 12)	t20	(25, 17)
t4	(11, 8)				

TABLE VII: A PARETO-OPTIMAL SCHEDULE FOR FIG. 3 (SCHEDULE RESULT = (27, 14))

Fire transition	Costs point	Fire transition	Costs point	Fire transition	Costs point
t15	(0, 0)	t13	(6, 5)	t4	(21, 9)
t11	(0, 0)	t14	(11, 6)	t19	(21, 10)
t1	(0, 0)	t17	(11, 7)	t5	(23, 10)
t12	(2, 3)	t18	(15, 8)	t20	(23, 13)
t2	(3, 3)	t3	(15, 8)	t6	(27, 14)
t16	(6, 3)				

V. CONCLUSIONS

Practically, we usually need to consider multiple criteria in the AMS scheduling, such as the minimal makespan, the least money, and the least tardiness. This paper proposes a multiobjective A∗ search approach within the reachability graphs of Petri nets of AMSs. It can obtain all Pareto-optimal schedules when an admissible heuristic function is adopted.

In the future, we will research on how to design more-informed heuristic functions for the proposed method. In addition, how to speed up the search process by using some relaxation strategies for large-scale systems also interests us.

CONFLICT OF INTEREST

The authors declare no conflict of interest

AUTHOR CONTRIBUTIONS

Chong Yu, Bo Huang concuted the research; Chong Yu, JianGen Hao analyzed the data; Bo Huang, JianGen Hao wrote the paper; all authors had approved the final version.

REFERENCES

[1] G. Tuncel and G. M. Bayhan, "Applications of Petri nets in production scheduling: A review," *Int. J. Adv. Manuf. Tech.*, vol. 34, no. 7-8, pp. 762-773, Oct. 2007.

[2] N. Wu and M. Zhou, "Deadlock resolution in automated manufacturing systems with robots," *IEEE Trans. Autom. Sci. Eng.*, vol. 4, no. 3, pp.474-480, Jul. 2007.

[3] Z. Li and M. Zhou, *Deadlock Resolution in Automated Manufacturing Systems: A Novel Petri Net Approach*, London, U.K.: Springer, 2009.

[4] B. Huang, M. Zhou, P. Zhang, and J. Yang, "Speedup techniques for multiobjective integer programs in designing optimal and structurally simple supervisors of AMS," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 48, no. 1, pp. 77-88, Jan. 2018.

[5] B. Huang, M. Zhou, Y. Huang, and Y. Yang, "Supervisor synthesis for FMS based on critical activity places," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 5, pp. 881-890, May 2019.

[6] D. Y. Lee and F. D. Cesare, "Scheduling flexible manufacturing systems using Petri nets and heuristic search," *IEEE Trans. Robot. Autom.*, vol. 10, no. 2, pp. 123-132, Apr. 1994.

[7] H. H. Xiong and M. Zhou, "Scheduling of semiconductor test facility via Petri nets and hybrid heuristic search," *IEEE Trans. Semicond. Manuf.*, vol. 11, no. 3, pp. 384-393, Aug. 1998.

[8] J. Luo, K. Xing, M. Zhou, X. Li, and X. Wang, "Deadlock-free scheduling of automated manufacturing systems using Petri nets and hybrid heuristic search," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 45, no. 3, pp. 530-541, Mar. 2015.

[9] G. Mej´ıa and K. Ni˜no, "A new hybrid filtered beam search algorithm for deadlock-free scheduling of flexible manufacturing systems using Petri nets," *Comput. Ind. Eng.*, vol. 108, pp. 165-176, 2017.

[10] H. Lei, K. Xing, L. Han, and Z. Gao, "Hybrid heuristic search approach for deadlock-free scheduling of flexible manufacturing systems using Petri nets," *Appl. Soft Comput.*, vol. 55, pp. 413-423, 2017.

[11] S. Peng, T. Li, J. Zhao, Y. Guo, S. Lv, G. Z. Tan, and H. Zhang, "Petri net-based scheduling strategy and energy modeling for the cylinder block remanufacturing under uncertainty," *Robot. Com-Int. Manuf.*, vol. 58, pp. 208-219, 2019.

[12] B. Huang, Y. Sun, Y. Sun, and C. Zhao, "A hybrid heuristic search algorithm for scheduling FMS based on Petri net model," *Int. J. Adv. Manuf. Tech.*, vol. 48, no. 9-12, pp. 925-933, Jun. 2010.

[13] B. Huang, X. Shi, and N. Xu, "Scheduling FMS with alternative routings using Petri nets and near admissible heuristic search," *Int. J. Adv. Manuf. Tech.*, vol. 63, no. 9-12, pp. 1131-1136, Dec. 2012.
[14] B. Huang, R. Jiang, and G. Zhang, "Search strategy for scheduling flexible manufacturing systems simultaneously using admissible heuristic functions and nonadmissible heuristic functions," *Comput. Ind. Eng.*, vol. 71, pp. 21-26, May 2014.
[15] J. Ezpeleta, J. M. Colom, and J. Martinez, "A Petri net based deadlock prevention policy for flexible manufacturing systems," *IEEE Trans. Robot. Autom.*, vol. 11, no. 2, pp. 173-184, Apr. 1995.
[16] F. Tricas, F. Garcia-Valles, J. Colom, and J. Ezpeleta, "An iterative method for deadlock prevention in FMS," *Discrete Event Systems*, Springer, 2000, pp. 139-148.
[17] J. Ezpeleta, F. Tricas, F. Garcia-Valles, and J. M. Colom, "A banker's solution for deadlock avoidance in FMS with flexible routing and multiresource states," *IEEE Trans. Robot. Autom.*, vol. 18, no. 4, pp. 621-625, Aug. 2002.
[18] B. Huang, M. Zhou, G. Zhang, A. C. Ammari, A. Alabdulwahab, and A. G. Fayoumi, "Lexicographic multiobjective interger programming for optimal and structurally minimal Petri net supervisors of automated manufacturing systems," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 45, no. 11, pp. 1459-1470, Nov. 2015.
[19] B. S. Stewart and C. C. White III, "Multiobjective A*," *Journal of the ACM (JACM)*, vol. 38, no. 4, pp. 775-814, 1991.
[20] W. Zeng and R. L. Church, "Finding shortest paths on real road networks: the case for A," *International journal of Geographical Information Science*, vol. 23, no. 4, pp. 531-543, 2009.

Chong Yu received the B.S. degree in network engineering from Zhejiang University of Technology, Hangzhou, China, in 2017. He will get the M.S. degree in computer technology from Nanjing University of Science and Technology, Nanjing, China, in 2020.

His research interests include Petri nets, discrete event systems, automated manufacturing systems, heuristic search, and intelligent transport systems.

Bo Huang received his B.S. and Ph.D. degrees from Nanjing University of Science and Technology, Nanjing, China, in 2002 and 2006, respectively.

He joined Nanjing University of Science and Technology (NUST), China, in 2007, where he is currently a full professor with the School of Computer Science and Engineering. He was a visiting professor at the University of Missouri - Kansas City, MO, USA, in 2013, and a visiting scholar at the New Jersey Institute of Technology, Newark, NJ, USA, from 2014 to 2015. His research interests include intelligent automation, Petri nets, symbolic computation, transportation and multi-robot systems. He has published 50+ papers in these areas.

Dr. Huang has served in the program committee for IEEE SMC 2018, IEEE ICNSC 2018, IEEE SMC 2017, IEEE ICNSC 2017, ICMSCD 2017, ICEEAC 2017, EEA 2016. He has been serving as a reviewer for Automatica, the IEEE Transactions on Systems, Man, and Cybernetics: Systems, the IEEE Transactions on Automation Science and Engineering, the IEEE Transactions on Industrial Informatics, the IEEE Transactions on Intelligent Transportation Systems, Information Sciences, IET Control Theory and Application, ISA Transactions, the International Journal of Advanced Manufacturing Technology, the Enterprise Information Systems, etc.

JianGen Hao received the B.E. degree in thermal automatic control and information technology from Beihang University, Beijing, China, in 2006. He received the M.E. and Ph.D. degrees from Beijing Jiaotong University, Beijing, China, in 2008 and 2014, respectively.

He joined Nanjing LES Information Technology Co., LTD, Nanjing, China in 2013, and is now a senior engineer. His research interests include the model free adaptive control, learning control and intelligent traffic control.