第 27 卷第 2 期
2010 年 2 月

控 制 理 论 与 应 用
Control Theory & Applications

Vol. 27 No. 2
Feb. 2010

Article ID: 1000−8152(2010)02−0199−06

Reconfigurable manufacturing cell error recovery: an approach based on improved net rewriting systems

LI Jun[1], DAI Xian-zhong[1], MENG Zheng-da[1], DOU Jian-ping[2]

(1. School of Automation, Southeast University, Nanjing Jiangsu 210096, China;

2. School of Mechanical Engineering, Southeast University, Nanjing Jiangsu 210096, China)

Abstract: The strategy and method for recovering an error occurring in a reconfigurable manufacturing cell(RMC) are proposed by a partial and temporary modification on the supervisor in Petri net(PN) formalism for normal operation control. Firstly, the improved net rewriting system (INRS) is presented that offers a direct way to dynamical structural changes in a PN model. Subsequently, an INRS-based error recovery method is presented, where the INRS is used to operate a PN supervisor and induct it into a correct state from the error one. After error recovery the structure and expected properties of a resulting supervisors is naturally preserved. Finally, with help of an example, the error recovery method presented is illustrated and the result shows the validity of the method.

Key words: error recovery; Petri nets; net operation; reconfigurable manufacturing system; supervisory control

CLC number: TH165 **Document code:** A

基于改进的网重写系统途径的可重构制造单元故障恢复

李 俊[1], 戴先中[1], 孟正大[1], 窦建平[2]

(1. 东南大学 自动化学院, 江苏 南京 210096; 2. 东南大学 机械工程学院, 江苏 南京 210096)

摘要: 提出了一种用于可重构制造单元故障恢复的策略与方法, 允许故障发生时, 通过对用于正常操作控制的Petri网形式的监督控制器进行局部、临时性的修改, 实现故障的恢复. 首先, 提出改进的网重写系统, 可用于动态改变Petri网模型结构. 然后,提出了基于改进的网重写系统的故障恢复方法, 其中改进的网重写系统用于操作、引导Petri网监督控制器由错误状态进入正确状态. 故障恢复后监督控制器的结构与期望属性维持不变. 最后, 以实例演示了该故障恢复方法的应用, 证实了方法的有效性.

关键词: 故障恢复; Petri网; 网运算; 可重构制造系统; 监督控制

1 Introduction

Reconfigurable manufacturing systems(RMS) are designed for rapidly adapting to frequent market changes in today's manufacturing environment[1]. The advent of RMS has given rise to the need for reconfigurability and rapid responsiveness of their supervisory controllers in response to both changes in the RMS configuration caused by the changing market and disturbances by errors. The issue of Petri net(PN) supervisor synthesis and reconfiguration has been researched in our previous work[2]. However, the issue of error recovery in RMS has hardly been discussed in the literature. This paper intends to present a novel strategy and method of error recovery in RMS.

Graphic methods are one of the most competitive ways to deal with errors. For instance, several integrated error handling methods using graphic formalisms were presented in [3∼5]. Among them the handling codes for all possible errors are integrated into the supervisory controllers of the controlled plants at the construction stage. However, the recovery logic for some errors is left unused or no recovery logic is prepared for errors that have not been anticipated. It is worth mentioning that, Zhou and DiCesare[6] provided a strategy where only the most common errors are considered at the controller design stage and the remainder are handled by augmenting a controller in the run-time of the controlled system. However, with increasing occurrences of errors in the system, the original controller will become more and more colossal and complicated.

Received date: 2009−06−23; Revised date: 2009−09−20.

Foundation item: supported in part by the China Postdoctoral Science Foundation Funded Project(20080431052).

This paper proposes a novel error recovery strategy and method where the recovery logic in response to an error occurring in a RMS is considered as a partial and temporary modification or adjustment on the PN-based supervisor for normal operation control. The net modification or adjustment is performed by improved net rewriting systems(INRS).

2 Improved net rewriting systems(INRS)

2.1 Definition of INRS

In contrast with net rewriting system[7], the improved net rewriting system presented by Li et al.[8] can not only maintain the ability for dynamically changing the structure of a PN but also preserve its original behavioral properties. In this section, we present a reduced version of INRS.

Definition 1 An improved net rewriting system (INRS) is a triple-tuple $\mathscr{N} = (\mathscr{G}, \mathscr{R}, \mathscr{L})$, where $\mathscr{G} = (P, T, F, M_0)$ is an underlying PN for rewriting; $\mathscr{R}$ is a finite set of net rewriting rules; $\mathscr{L}$ is a general net block type library.

1) For any $r \in \mathbb{R}$, r is expressed as $r = (L, R, {}^{\cdot}\tau, \tau^{\cdot})$, where L is a net block in $\mathscr{G}$, called the left-hand side of r, R is used to rewriting L, called the right-hand side of r, and ${}^{\cdot}\tau(\tau^{\cdot})$ is the input (output) interface relations between L and R used for locating during net rewriting. The types of L and R are restricted to $\mathscr{L}$. Three net block classes are taken from $\mathscr{L}$ and defined in Section 2.2.

2) Applying the rule r to $\mathscr{G}$, a new PN $\mathscr{G}' = (P', T', F', M_0')$ is obtained, where$P' = P \cup P_{\rm R} - P_{\rm L}$, $T' = T \cup T_{\rm R} - T_{\rm L}$, F' and M_0' are given by the following Eqs.(1) and (2), respectively:

$$F'(x,y) = \begin{cases} F(x,y), & x,y \notin T_{\rm R} \cup P_{\rm R}, \\ F_{\rm R}(x,y), & x,y \in T_{\rm R} \cup P_{\rm R}, \\ \dfrac{\sum\limits_{y_i \in {}^{\cdot}\tau y} F(x,y_i)}{|{}^{\cdot}\tau y|}, & x \notin T_{\rm R} \cup P_{\rm R},\ y \in T_{\rm R} \cup P_{\rm R}, \\ \dfrac{\sum\limits_{x_i \in \tau^{\cdot} x} F(x_i,y)}{|\tau^{\cdot} y|}, & x \in T_{\rm R} \cup P_{\rm R},\ y \notin T_{\rm R} \cup P_{\rm R}. \end{cases} \tag{1}$$

$$M_0'(p) = \begin{cases} M_0(p), & p \notin P_{\rm R}, \\ (M_{\rm R})_0(p), & p \in P_{\rm R}. \end{cases} \tag{2}$$

For more details, readers refer to [8].

2.2 General net block classes

Three net block classes that are included in $\mathscr{L}$ and used in this paper are introduced below.

Definition 2 Given a Petri net block $\mathscr{G} = (P, T, F, M_0)$, $\mathscr{G}$ is called a single input and single output marked graph (S^{2}MG), if the following conditions are satisfied:

1) $\mathscr{G}$ is a marked graph, i.e., $\forall p \in P$, $|{}^{\cdot}p| = |p^{\cdot}| = 1$;

2) $\exists t_i, t_j \in T (i \neq j)$, such that ${}^{\cdot}t_i = t_j^{\cdot} = \emptyset$, $\forall t \in T - \{t_i\}$, t is not a source, i.e., ${}^{\cdot}t \neq \emptyset$, and $\forall t \in T - \{t_j\}$, t is not a sink, i.e., $t^{\cdot} \neq \emptyset$;

3) There exists at least one elementary path from t_i to $t_j$without marked places and there exists at most one marked place appearing in the elementary path from t_i to t_j;

4) Every circuit including $\mathscr{G}$ has at least one marked place.

t_i and t_j are the input and output nodes of $\mathscr{G}$, i.e., $t_{\rm in} = t_i$ and $t_{\rm out} = t_j$, respectively.

A complex S^{2}MG as an example is shown in Fig.1(a).

Definition 3 Given a Petri net block $\mathscr{G} = (P, T, F, M_0)$, $\mathscr{G}$ is called a single input and single output state machine(S^{2}SM), if the following conditions are satisfied:

1) $\mathscr{G}$ is a state machine, i.e., $\forall t \in T$, $|{}^{\cdot}t| = |t^{\cdot}| = 1$;

2) $\exists p_i, p_j \in P (i \neq j)$, such that ${}^{\cdot}p_i = p_j^{\cdot} = \emptyset$, $\forall p \in P - \{p_i\}$, p is not a source, i.e., ${}^{\cdot}p \neq \emptyset$, and $\forall p \in P - \{p_j\}$, p is not a sink, i.e., $p^{\cdot} \neq \emptyset$;

3) $\forall p \in P$, p is not marked, i.e., $M_0(p) = 0$.

Let p_i and p_j be the input and output nodes of $\mathscr{G}$, i.e., $p_{\rm in} = p_i$ and $p_{\rm out} = p_j$, respectively.

An example S^{2}MG is shown in Fig.1(b).

Definition 4 Given a Petri net block $\mathscr{G} = (P, T, F, M_0)$, $\mathscr{G}$ is called a non-strict S^{2}SM(NS^{2}SM), if it meets:

1) $\exists p_i, p_j \in P$ $(i \neq j)$, such that ${}^{\cdot}p_i = p_j^{\cdot} = \emptyset$, $\forall p \in P - \{p_i\}$, p is not a source, i.e., ${}^{\cdot}p \neq \emptyset$, and $\forall p \in P - \{p_j\}$, p is not a sink, i.e., $p^{\cdot} \neq \emptyset$;

2) There exist $m(\geqslant 1)$ S^{2}MG components in $\mathscr{G}$, denoted as $C_1, C_2, \cdots, C_m$, respectively. Let $C_i = (P_i, T_i, F_i, M_{i0})$, where $1 \leqslant i \leqslant m$;

3) $\forall t \in T - \bigcup\limits_{1}^{m} T_i$, $|{}^{\cdot}t| = |t^{\cdot}| = 1$;

4) $\mathscr{G}$ changes into an S^{2}SM by reducing all S^{2}MG components to single transitions.

Let p_i and p_j be the input and output nodes of $\mathscr{G}$, i.e., $p_{\rm in} = p_i$ and $p_{\rm out} = p_j$, respectively.

An NS^{2}SM can be regarded as an S^{2}SM into which S^{2}MG blocks are embedded. An example NS^{2}SM is given in Fig.1(c).

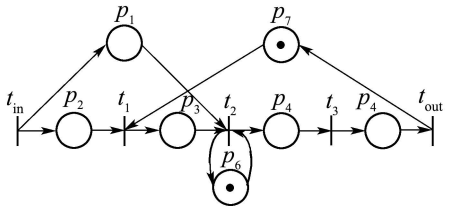

Fig. 1(a) An S^2MG

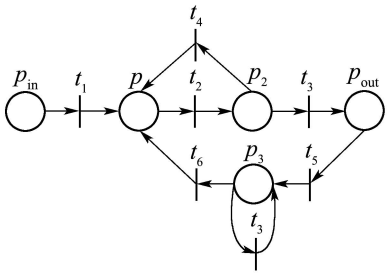

Fig. 1(b) An S^2SM

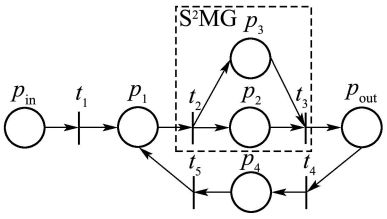

Fig. 1(c) A non-strict S^2SM embedded with an S^2MG

The defined three net block types can serve as basic components for building subnets and net paths for error recovery in Petri net controllers.

2.3 Basic theoretical results of INRS

A theorem supporting INRS-based error recovery is presented below.

Theorem 1 Given a Petri net $\mathscr{G} = (P, T, F, M_0)$, in which exists an S^2SM component or an NS^2SM component, $\mathscr{G}_s$, whose input and output nodes are p_{in}, p_{out}, respectively. If p_{in} obtains a token, then the number of tokens contained in $\mathscr{G}_s$ keeps 1 despite occurrence of any transition firing sequence, and there must exist one transition firing sequence that can make p_{out} get a token.

The proof is omitted here. This theorem guarantees that the expected states will not be blocked and will be free of deadlock during operations of an INRS if the error recovery logic is expressed by S^2SM or NS^2SM net, namely, a system can be inducted normally from an error state to a restart state via the recovery logic.

3 INRS-based error recovery method

3.1 Principles of error recovery

We propose here a strategy to separate recovery logic in response to the errors occurring in an RMS from normal operation control logic. In return, design of PN controllers for normal operation control and design of error recovery logic can be carried out in a parallel manner. Furthermore, recovery logic of an error is treated as a partial and temporary modification or adjustment to the controller with normal operation control logic when the error occurs.

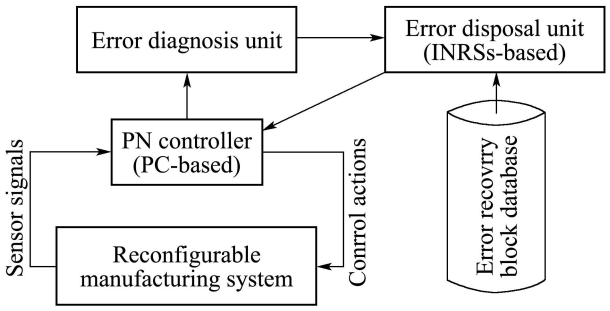

Fig. 2 INRS-based error recovery architecture for an RMS

An architecture for a Petri net supervisory control system incorporated with error detection, diagnosis (e.g., the method presented in [9]) and recovery is shown in Fig. 2. In a Petri net controller[2], places execute normal operation control logic (output control actions), while transitions detect and read sensor signals. A transition associates with a firing condition that is a Boolean expression consisting of several sensory signal variables. An enabled transition will fire once its firing condition is true. In the architecture: 1) faults or errors occurring in a RMS are detected by transitions. An error detection function defined for a transition will return a value for further error diagnosis when an error is occurring; 2) the error diagnosis unit judges the error type from the detection value sent by the controller by searching a fault tree. Further, a policy and a path for recovery of the error are decided in the same unit; 3) the error disposal unit executes error recovery, where a net block for recovering the error, corresponding to the decided policy and path, is token from the error recovery block database, firstly; then the block is operated by an INRS to rewrite the Petri net controller to recover the error. All error recovery blocks pertain to the types of S^2SM and NS^2SM defined in Section 2.2.

From INRS, two net operations of addition and subtraction are defined below to accelerate net rewriting.

Definition 5 1) A net addition is an INRS rewriting process in which a net block B is added to the underlying PN A of an INRS $\mathscr{N}$ and the interface relations for locating are $^{\cdot}\tau$ and $\tau^{\cdot}$, denoted as $A + B/$

$({}^{\cdot}\tau, \tau^{\cdot})$; 2) A net subtraction is an INRS rewriting process in which a net block B is removed from the underlying PN A of an INRS $\mathcal{N}$ and the interface relations for locating are ${}^{\cdot}\tau$ and $\tau^{\cdot}$, denoted as $A - B/({}^{\cdot}\tau, \tau^{\cdot})$.

From Theorem 2, this corollary can be deduced straightforwardly.

Corollary 1 The net $A + B/({}^{\cdot}\tau, \tau^{\cdot})(A - B/({}^{\cdot}\tau, \tau^{\cdot}))$ is live, bounded, and reversible, if the net A is live, bounded, and reversible, and the block B belongs to the types of S^2SM and NS^2SM.

This corollary supports directly the method of error recovery with the INRS approach.

3.2 Error recovery polices

For errors that can be recovered automatically, perhaps, the most effective error recovery policies or trajectories are these presented by Zhou and DiCesare[6], i.e. input conditioning, backward recovery, and forward error recovery. However, for a complex system, not all errors can be anticipated and not all errors can be recovered automatically. Sometimes human intervention is necessary, which is usually neglected in the literature. Therefore, we provide another policy for error recovery, namely the human intervention policy. The human intervention policy is used to recover error when the three automatic error recovery policies fail to handle an unknown error occurring in the system. The former three policies are illustrated in [6]. As for the last one, it is similar to them and the only difference is in that the recovery path and the restart state are not prescribed but set by an operator on the spot.

3.3 Main procedures for error recovery

Procedure 1 Once a transition in a PN controller detects an error occurring, the diagnosis unit will judge the error type. Meanwhile, the disposal unit will prevent immediately the state evolution of the controller, and get the corresponding PN block for recovery of the error from the block database.

Procedure 2 The disposal unit constructs automatically an INRS. In the INRS, the PN controller, denoted as A, is regarded as its underlying Petri net, the error recovery net block, denoted as B, is considered as the added net. The input and output interfaces of A are the error point (a place) and the restart point prescribed (a place), respectively. The input and output interfaces of B are its beginning and end, respectively. Executing the net addition, $A + B/({}^{\cdot}\tau, \tau^{\cdot})$, where $({}^{\cdot}\tau, \tau^{\cdot})$ is the interface relation for addition decided by the input and output interfaces of A and B, one will obtain an intermediate PN controller C.

Procedure 3 Run the added PN logic for error recovery until the restart point is reached. At the moment, the error is recovered.

Procedure 4 The disposal unit deletes the added net block for error recovery. Here an INRS is constructed. In the INRS, the existing PN controller model C is regarded as its underlying Petri net, and a part of the added error recovery net $(B - {}^{\cdot}\tau B - \tau^{\cdot} B)$ is considered to be the removed net. Similar to the second procedure, the interface relation for the operation can be determined, denoted as $({}^{\cdot}\tau', \tau'^{\cdot})$. After executing the net subtraction, $C - (B - {}^{\cdot}\tau B - \tau^{\cdot} B)/({}^{\cdot}\tau', \tau'^{\cdot})$, the original Petri net controller with a new error-free state is obtained.

Procedure 5 Restart the controller to execute the control logic for normal operation of the RMS from the restart state.

The presented method for error recovery is partial and temporary. With the exception of the transitorily dynamic stage of error recovery, the size or complexity of the controller with control logic for normal operations is kept unchanged.

4 Illustrative example

Given PN control logic for an automated guided vehicle AGV1's task 1, shown in Fig. 3(a). It is intercepted from the PN controller G_1 for a reconfigurable manufacturing cell (see [2]). The interpretation of places and transitions in the PN controller for AGV1 is shown in Table 1.

Table 1 Meanings of nodes in PN control logic for AGV1's task 1

Place/transition	Meanings
$A11.p0$	AGV1 available
$A11.p1$	AGV1 moving to loading station
$A11.p2$	Part available in pickup point
$A11.p3$	AGV1 being loaded a part
$A11.p4$	AGV1 moving to delivery point
$A11.p5$	Delivery point available
$A11.p6$	AGV1 being unloaded the part
$A11.t1 \sim 5$	Start or end of an operation

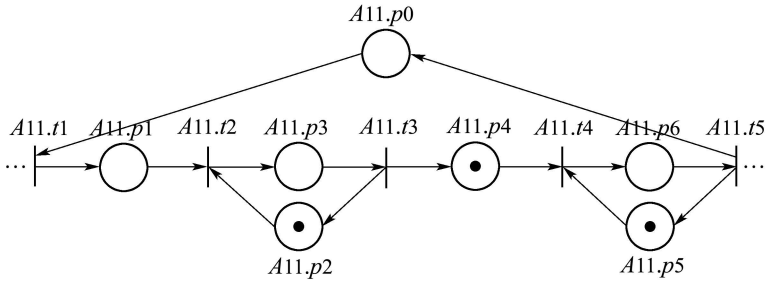

Fig. 3(a) Task A11 control block intercepted from PN controller G_1

Assume that AGV1's control logic is executing and at one moment, the carried part slides from AGV1 that is detected by the transition $A11.t4$. By executing stepwise the main procedures of error recovery, as shown in Fig. 3(a)∼(d), the original Petri net controller with a new error-free state($A11.p1$) is obtained. Then restart the controller to execute the control logic for normal operation of the RMS from the restart state.

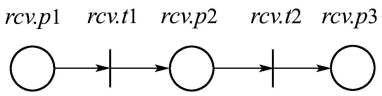

Fig. 3(b) Error recovery block B

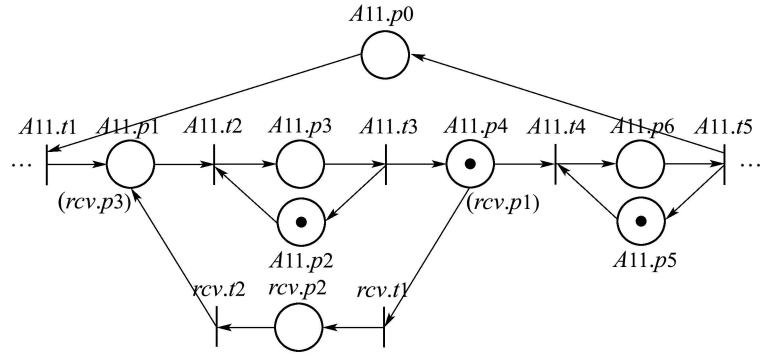

Fig. 3(c) Resultant controller C obtained by net addition $G_1 + B/((A11.p4, rcv.p1), (A11.p1, rcv.p3))$

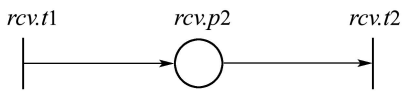

Fig. 3(d) Net block B'

The result indicates that using the presented method, error recovery reflects eventually changes in the state of the controller and the original configuration and basic behavioral properties, i.e., liveness, boundedness (safeness) and reversibility, of the original controller are preserved.

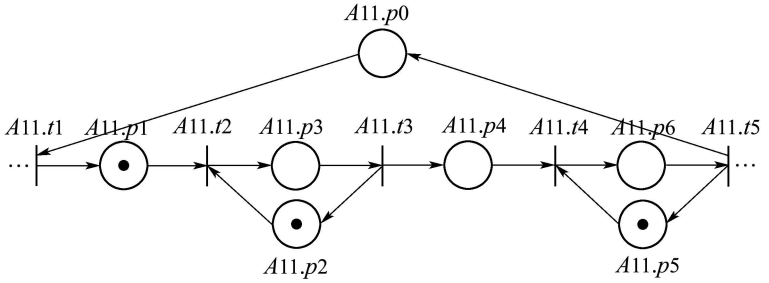

Fig. 3(e) Resultant controller with an error-free state by net subtraction $G_1 + B/((A11.p4, rcv.p1), (A11.p1, rcv.p3))$

5 Conclusion

This paper proposed the strategy in which control logic of normal operations and recovery logic, in response to the disturbance of errors, are treated apart. Following the strategy, a method for error recovery based on INRS has been presented subsequently. In

the method, error recovery logics modeled by PN blocks are operated by INRS to induct PN supervisors of RMS into error-free restart states from error states and cancelled later. Noticeably, the advantage of the error recovery method is that the structure and important behavioral properties of the original controllers are kept unchanged.

References:

[1] KOREN Y, HEISEL U, JOVANE F, et al. Reconfigurable manufacturing systems[J]. *CIRP Annals*, 1999, 48(2): 527 – 540.

[2] LI J, DAI X, MENG Z. Automatic reconfiguration of Petri net controllers for reconfigurable manufacturing systems with an improved net rewriting system-based approach[J]. *IEEE Transactions on Automation Science & Engineering*, 2009, 6(1) : 156 – 167.

[3] PARK E, TILBURY D M, KHARGONEKAR P P. A modeling and analysis methodology for modular logic controllers of machining systems using Petri nets formalism[J]. *IEEE Transactions on Systems, Man, Cybern- Part C: Application and Reviews*, 2001, 31(2) : 168 – 186.

[4] SHAH S S, ENDSLEY E W, LUCAS M R, et al. Reconfigurable logic control using modular FSMs: design, verification, implementation, and integrated error handling[C] //*Proceedings of American Control Conference*. Piscataway: IEEE, 2002, 8: 4153 – 4158.

[5] WU H, JOSHI S B. Error recovery in MPSG-based controllers for shop floor control[C] //*Proceedings of IEEE International Conference on Robotics and Automation*. Piscataway: IEEE, 1994: 1374 – 1377.

[6] ZHOU M, DICESARE F. Adaptive design of Petri net controllers for error recovery in automated manufacturing systems[J]. *IEEE Transactions on Systems, Man, and Cybernetics*, 1989, 19(5): 963 – 973.

[7] LLORENS M, OLIVER J. Structural and dynamic changes in concurrent systems: reconfigurable Petri nets[J]. *IEEE Transactions on Computer*, 2004, 53(9): 1147 – 1158.

[8] LI J, DAI X Z, MENG Z D, et al. Rapid design and reconfiguration of Petri net models for reconfigurable manufacturing cells with improved net rewriting systems and activity diagrams[J]. *Computers & Industrial Engineering*, 2009, 57(4): 1431 – 1451.

[9] XUE F, ZHENG D. Fault diagnosis of time event graph[J]. *Control Theory & Applications*, 2005, 22(4): 609 – 614.

LI Jun was born in Anhui Province. He received the Ph.D. degree in control theory and control engineering from Southeast University, Nanjing, China, in 2007. Now he is performing a postdoctoral research project of reconfigurable manufacturing systems. His current research interests include Petri net theory, modeling, simulation, and supervisory control of discrete event systems, and robotics. E-mail: j.li@seu.edu.cn;

DAI Xian-zhong was born in Jiangsu Province. He received the Ph.D. degree in electrical engineering from Tsinghua University, Beijing, China, in 1986. He is currently a professor in the School of Automation, Southeast University. His work has been in power system control, artificial neural networks, robotics, and advanced manufacturing systems;

MENG Zheng-da was born in Jiangsu Province. He received the M.S. degree in control theory and control engineering from Southeast University, Nanjing, China, in 1988. He is currently a professor in the School of Automation, Southeast University. His main research interest is robotic control.

DOU Jian-ping was born in Hunan Province. He received the Ph.D. degree in control theory and control engineering from Southeast University, Nanjing, China, in 2009. He is with the School of Mechanical Engineering, Southeast University. His current research interests include modeling, optimization, and control of manufacturing systems, and mechatronics.