Computers & Industrial Engineering 83 (2015) 139–150

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Robust supervision using shared-buffers in automated manufacturing systems with unreliable resources ☆

Hao Yue [a,b,c], Keyi Xing [b], Hesuan Hu [d], Weimin Wu [a,*], Hongye Su [a]

[a] The State Key Laboratory of Industrial Control Technology and the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, Zhejiang 310027, PR China
[b] The State Key Laboratory for Manufacturing Systems Engineering and the Systems Engineering Institute, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, PR China
[c] Lab of Granular Computing, Minnan Normal University, Zhangzhou, Fujian 363000, PR China
[d] School of Electro-Mechanical Engineering, Xidian University, Xi'an, Shaanxi 710071, PR China

ARTICLE INFO

Article history:
Received 6 October 2013
Received in revised form 11 January 2015
Accepted 29 January 2015
Available online 9 February 2015

Keywords:
Deadlock avoidance
Robust control
Automated manufacturing systems
Control policy

ABSTRACT

It has been an active area of research to solve the modeling, analysis, and deadlock control problems for automated manufacturing systems (AMSs). So far, all the system resources are assumed to be reliable in most of the existing approaches for deadlock-free and nonblocking supervisory control. However, many resources of AMSs are subject to failure in the real world. In order to develop a more practical and applicable supervisor, this work takes into consideration of multiple unreliable resources in a class of AMSs. On the basis of two variants of Banker's Algorithm, this paper presents a robust supervisory control policy to avoid deadlock and blocking in these systems. The policy tries to make the best use of buffers of the shared resources to achieve the control objectives. Our controller is qualified to handle simultaneous multi-resource failures. By using formal language and automata theory, we establish its correctness. Moreover, our proposed method is verified via an AMS example, and we make comparison studies between our policy and some of the other similar type of policies in the literature.

1. Introduction

Automated manufacturing systems (AMSs), which have evolved for decades, are computer-controlled production facilities adaptable to variable production plans and goals. Recently, modern information technology has been increasingly and extensively applied to AMSs research. On the basis of the up-to-date technology in information science and engineering, researchers addressed a number of issues regarding modeling, simulation, control, and scheduling for AMSs (Ahmad, Huang, & Wang, 2011; Choi & Ko, 2009; Ferrarini & Piroddi, 2008; Hsueh, 2010; Huang, Pan, & Zhou, 2012; Huang, May, Wu, & Huang, 2013; Jeng, Xie, & Peng, 2002; Roszkowska & Reveliotis, 2013; Wang & Wu, 1998; Wu, Zhou, & Li, 2008; Xing, Han, Zhou, & Wang, 2012; Yalcin & Namballa, 2005). As a consequence, this field develops rapidly, and manufacturers can benefit much in reducing cost, increasing productivity, and improving products' quality so as to meet the global market competition.

Supervisory control is often treated in a logical domain. It aims to constrain system behavior to the legal scope (including deadlock-free zones). Therefore, supervisory control is one of the fundamental needs to establish and maintain AMSs in manufacturing engineering theory and applications. Generally, control logic is formally modeled by using formalisms such as automata and Petri nets. With the help of the previously set up system model, system designers or control engineers develop control software to meet all kinds of desired requirements such as deadlock-free operation. Finally, executable control codes are generated and the control goals are fulfilled. Actually, the whole procedure demonstrates a typical successful application of state-of-the-art supervisory control theory to contemporary manufacturing systems.

AMSs usually exhibit high degrees of process concurrency, resource sharing, and resource competition. The resources in AMSs include computer numerically controlled machine tools, workstations, fixtures, robots, automated guided vehicles, and other material-handling devices. Various parts entering the system compete for limited resources. As a result, this may in turn lead to dead states or deadlocks (Fanti & Zhou, 2004; Li, Wu, & Zhou, 2012) under improper control or supervision. Deadlock is a highly

☆ This manuscript was processed by Area Editor Manoj Tiwari.

* Corresponding author at: The State Key Laboratory of Industrial Control Technology and the Institute of Cyber-Systems and Control, Zhejiang University, No. 38, Zheda Road, Hangzhou, Zhejiang 310027, PR China. Tel.: +86 571 87952369.

E-mail addresses: yuehao1980@mail.xjtu.edu.cn (H. Yue), kyxing@sei.xjtu.edu.cn (K. Xing), huhesuan@gmail.com (H. Hu), wmwu@iipc.zju.edu.cn (W. Wu), hysu@iipc.zju.edu.cn (H. Su).

http://dx.doi.org/10.1016/j.cie.2015.01.028
0360-8352/

unfavorable situation in which a set of parts are requesting or waiting for the resources held by other parts in the same set (Fanti & Zhou, 2004). In deadlock situations, the whole system or a part of it remains indefinitely blocked due to circular waiting. Furthermore, deadlocks may give rise to catastrophic results in some AMSs such as semiconductor manufacturing systems. Therefore, trying to resolve deadlock issues in AMSs has received significant attention, and there have been a large number of research achievements (Chao, 2012; Chen, Li, & Al-Ahmari, 2013; Chu & Xie, 1997; Cordone & Piroddi, 2013; Esther Cano, Rovetto, & Colom, 2012; Ezpeleta & Valk, 2006; Hu, Zhou, Li, & Tang, 2013; Huang, Shi, & Xu, 2012; Lewis, Gurel, Bogdan, Doganalp, & Pastravanu, 1998; Liu, Xing, Zhou, Han, & Wang, 2014; Nazeem & Reveliotis, 2012; Park & Reveliotis, 2001; Piroddi, Cordone, & Fumagalli, 2009; Reveliotis & Nazeem, 2013; Tricas & Ezpeleta, 2006; Uzam, Li, & Zhou, 2007; Wang, Wang, & Zhou, 2013; Xing, Zhou, Liu, & Tian, 2009; Zhang & Judd, 2008; Zhao & Hou, 2013).

At present, most of the deadlock control methods in the related literature do not consider the failure-prone or unreliable resources in AMSs. However, AMSs are often subject to resource failures, and additional blocking issues may arise due to resource breakdowns. Furthermore, there are various different kinds of resource faults such as tool breakages, hydraulic or electrical failures, pneumatic failures, sensor failures, and any other types of events that cause a resource to stop operating. Therefore, a deadlock resolution method must have a built-in function to accommodate unpredictable resource failures in real manufacturing systems. Unfortunately, few deadlock control policies in the existing literature can be applied directly to AMSs with unreliable resources. As a result, it has been a great challenge to develop robust or fault-tolerant supervisory control policies for deadlock resolution (Chew, Wang, & Lawley, 2011; Hsieh, 2011; Liu, Li, Barkaoui, & Al-Ahmari, 2013; Park & Lim, 1999; Yalcin, 2004; Yue, Xing, & Hu, 2014).

Considering failures both during processing mode and in the idle mode, Yalcin (2004) addresses the systematic procedures which reconfigure the supervisor to continue the operation of automated flexible manufacturing cells. By using finite-state automata as the formalism, the proposed supervisory controller accommodates resource failures as well as takes into consideration of deadlocks and dynamic routing flexibility options.

On the basis of non-ordinary Petri nets, Hsieh (2007, 2011) has made a sequence of significant contributions to the area of robustness analysis and fault-tolerant deadlock avoidance for flexible assembly/disassembly processes.

Liu et al. (2013) present exploratory research on the robustness of liveness-enforcing supervisors for AMSs in a Petri net formalism. Their work is intuitive and has the advantage of being able to avoid the reanalysis of the net. Moreover, the work bridges the gap between the existing deadlock control policies and their application to real-world systems with unreliable resources.

Lawley et al. concentrate on the robust supervisory control for deadlock avoidance and nonblocking control in a class of AMSs with unreliable resources (Chew & Lawley, 2006; Chew, Wang, & Lawley, 2009, 2011; Lawley & Sulistyono, 2002; Wang, Chew, & Lawley, 2008, 2009). Each resource discussed in their work is a workstation, which is different from the resources in most of the other work in the deadlock control research community. Specifically, a resource workstation addressed by Lawley et al. comprises a server or processor for processing parts, and a couple of units of buffer space for storing parts. For a given resource, the number of buffer space units is its capacity. Resource "failure" is only the failure of the server. When a resource fails, neither waiting part nor on-going part at the resource can be processed. Furthermore, each part type has its unique production route, and each operation stage is supported by only one resource. The supervisors proposed by Lawley et al. guarantee that the following desired requirement is always satisfied. In the face of unreliable resource's failures and repairs, every part type whose production does not need any of the failed resources can continue to produce smoothly without disruption.

Failure-dependent parts are those which will require unreliable resources in their future processing. The robust supervisory control policies considered by Lawley et al. are classified into two types (Wang, Chew, & Lawley, 2009). The first type of policies is referred to as absorbing type. They can clear all shared resources of failure-dependent parts so that the production of non-failure-dependent parts is not affected at all in case of resource failures. The second type of policies is named as distributing type. These policies locate failure-dependent parts by using the buffers of both shared resources and failure-dependent resources. To avoid repeating a large number of words and illustrations, we refer the readers to the work of Wang et al. (2009) for the detailed informal definitions and discussions about distributing type and absorbing type of policies. Furthermore, we try to present the formal definitions of the two policy types in Section 3.2 in this paper.

As implied in Wang et al. (2008, 2009), there is a lack of a robust supervisory control policy, of distributing type, that can handle simultaneous multi-resource failures. To the best of our knowledge, it is still an open problem to develop such a policy. We try to resolve this issue in this paper.

Enlightened by the work of Lawley et al. and others, this work proposes a robust supervisory control policy to avoid deadlock and blocking for a class of AMSs with multiple unreliable resources. By properly using buffer space units of the shared resources, our policy is of the so-called distributing type, and it can cope with the situation when more than one resource is likely to be down at the same time. We note that the notion "blocking" discussed in this paper is different from the classical one addressed in Ramadge and Wonham (1987). In fact, our "blocking" means that the failure of an unreliable resource propagates to destroy the smooth flow and production of part types not requiring the failed resource.

The work in this paper investigates the same class of AMSs as reported by Lawley et al. Moreover, we take into account the complicated scenario where multiple unreliable resources fail simultaneously. Our policy is a two-stage method that consists of two variants of Banker's Algorithm (BA). A state and its corresponding derived state are checked by the two variants of BA at two stages, respectively. The first stage is called 1-order BA. It is a test trying to lead the state to a new state where all resources except the failure-dependent ones have free buffer space. The second one is named 2-order BA. Taking the state output by the 1-order BA, it is a test that tries to change this state to another one where all resources have free buffer spaces. Our policy is illustrated to be of distributing type. In addition, we establish the policy correctness by using the classical theory of formal language and automata.

The rest of this paper is organized as follows. Section 2 reviews the fundamental concepts and properties with respect to the class of AMSs considered in this paper. In Section 3, we present a robust supervisory control policy for deadlock avoidance and nonblocking control in this class of AMSs. Moreover, an example is employed to show the effective execution of our policy. Section 4 presents the policy comparison studies. Finally, Section 5 concludes this paper.

2. Discrete event system model of an AMS with unreliable resources

This work focuses on the design of robust control policies for avoiding deadlock and blocking states in a class of AMSs with unreliable resources. This class of AMSs were initially researched in Chew and Lawley (2006). In this section, we review the AMSs and their main properties.