Available online at www.sciencedirect.com
SCIENCE DIRECT®
Robotics and Computer-Integrated Manufacturing 21 (2005) 346–354
Robotics and Computer-Integrated Manufacturing
www.elsevier.com/locate/rcim

An augmented Petri Net approach for error recovery in manufacturing systems control

Nicholas G. Odrey[a,*], Gonzalo Mejía[b]

[a]Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, USA
[b]Department of Industrial Engineering, Universidad de Los Andes, Bogotá, Colombia

Received 14 September 2004; received in revised form 10 November 2004; accepted 12 November 2004

Abstract

The construction of error recovery Petri subnets and similar representations have received considerable attention in the literature. Previous work has presented a multi-agent system representing various levels of control in a reconfigurable architecture. Agents pertaining to production, mediation, and error recovery within such an architecture were considered. Our focus here is on the workstation level of a hierarchy where the workstation has the capability for recovery from physical errors. The implications of error recovery tasks from the perspective of control are also discussed. The approach is based on integrating Petri subnet models within a general Petri Net model for a manufacturing system environment. In essence, the error recovery plan consists of a trajectory (Petri subnet) having the detailed recovery steps that are then incorporated into the workstation control logic. The logic is based on a Timed Petri Net model of the total production system. The Petri subset models consist of a sequence of steps required to reinstate the system back to a normal state. Once generated, the recovery subnet is incorporated into the Petri Net model of the original expected (error-free) model. Petri Net augmentations pertaining to various issues are discussed in detail throughout the paper. Issues include the implication of generated error recovery trajectories in the production activities, linking of production activity Net and the error recovery subnet, potential deadlocks, the role of resources, and part handling.

Keywords: Petri Nets; Error recovery; Flexible manufacturing systems; Manufacturing systems control

1. Introduction

In this paper, we focus on the workstation level of a hierarchical manufacturing system. A workstation is typically a set of parallel machines linked by material handling devices that perform one or more manufacturing and assembly operations. The workstation controller is the entity responsible for the coordination, execution and regulation of the activities at the physical workstation. The workstation controller receives a higher level command, generally form a higher level controller that issues a set of operations to be performed by the workstation with desired start and finish times. The workstation controller decomposes such a command into a lower level set of coordinated activities. In addition to executing activities, the workstation controller should also provide a reactive and adaptive response to errors and other disturbances [1].

Petri Nets have been successfully used for modeling and control the dynamics of flexible manufacturing systems. Several modeling approaches based on Petri Nets that include those of [1,2,3] have been proposed. Generally, the operations required on a part are modeled with combinations of places and transitions. The movement of tokens throughout the Net models the execution of the required operations. In this paper, we follow the modeling approach previously presented by Odrey and Ma [1] and Mejia and Odrey [4]. The Petri

*Corresponding author. Tel.: +1 610 758 4036; fax: +1 610 758 4886.
E-mail address: ngo0@lehigh.edu (N.G. Odrey).

0736-5845/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.rcim.2004.11.004

Net formalism can handle the complexities of highly detailed activities of a manufacturing workstation such as parallel machines, buffers of finite capacity, dual resources (multiple resources required simultaneously on one operation), alternative routings, and material handling devices to name a few.

The characteristics of physical error occurrence impose difficult challenges to the workstation controller. The controller must first handle simultaneously production and recovery activities, and second, errors that appear unexpectedly must be treated in real-time to avoid a dramatic decrease of performance. The error recovery problem has been extensively studied: Examples of automated reasoning systems for error recovery procedures, such as expert systems and neural networks have been proposed over the years and include [7,8,9].

Previous work pertained to addressing the issue of monitoring, diagnostics, and error recovery within the context of a hierarchical multi-agent system [5]. The system consists of production, mediator, and error recovery agents. Production agents contain both planner (scheduler) and control agents. Here, we address the error recovery agent within the hierarchical system at the workstation level in more detail. It is assumed that raw sensory information has been processed and is available. For complex systems the diagnostics task may be performed by a mediator agent. When an error is detected, the control agent diagnoses the error and requests the action of a recovery agent. In return, the recovery agent devises a plan to bring the system out of the error state. Such an error recovery plan consists of a trajectory having the detailed recovery steps that are incorporated into the control agent logic. In the context of Petri Nets, a recovery trajectory corresponds to a Petri subnet which models the sequence of steps required to reinstate the system back to a normal state. After being generated, the recovery subnet is incorporated into the workstation activities Net (the Petri Net of the multi-agent system environment). In this research, we follow the designation of others [3], and denote the incorporation of a recovery subnet into the activities Net as Net augmentation. The terms "original Net" or "activities Net" refer to the Petri Net representing the workstation activities (within a multi-agent environment) during the normal operation of the system. The Net augmentation brings several problems that require careful handling to avoid undesirable situations such as deadlocks.

2. Background

The construction of error recovery Petri subnets and similar representations is a topic which has received considerable attention in the literature. An abnormal state can become a normal state after other actions are finished or some conditions are met. Backward recovery suggests that a faulty state can become a normal state if an early stage in the original trajectory can be reached. The forward recovery trajectory consists of reaching a later. A forward trajectory is the most desirable but at the same time, it is the most difficult to implement with automated reasoning systems [6].

For example, [10] presented a typical Petri Net representation for machines breakdowns and alternative routings. Extensions to such model representations to handle a more complicated logic including requests for recovery actions and temporary storage in buffers were also accomplished [11]. This work was extended [12] with a representation to handle a more complicated logic that included requests for recovery actions and temporary storage in buffers.

Perhaps the most complete description of error recovery trajectories was developed by Zhou and DiCesare [3], who proposed three possible error recovery trajectories: input conditioning, backward error recovery and forward error recovery. The concept of input conditioning is that an abnormal state can become a normal state after other actions are finished or some conditions are met. Backward recovery suggests that a faulty state can become a normal state if an early stage in the original trajectory can be reached. The forward recovery trajectory consists of reaching a later state in the original trajectory after satisfying some operational constraints. Zhou and DiCesare [3] developed a formal description of these three possible trajectories in terms of Petri Net constructs. See Fig. 1 for an example. The input conditioning example shows a trajectory that "returns" to the state where the error occurred (Fig. 1a). The backward recovery trajectory aims to reach a state visited prior to error occurrence (Fig. 1b). Finally, the forward recovery trajectory aims to reach a state reachable from the state where

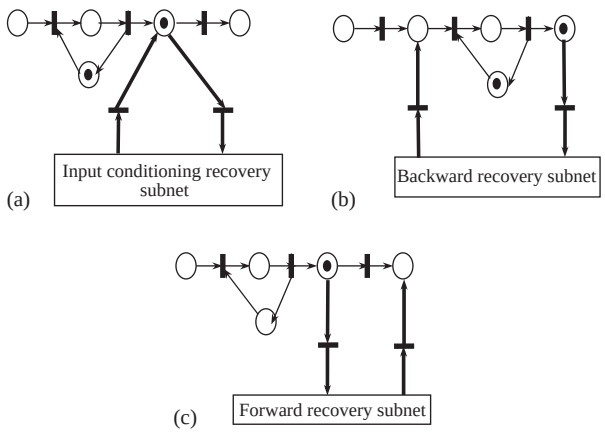

Fig. 1. Error Recovery Trajectories (following Zhou and DiCesare [3]): (a) example of input conditioning, (b) example of backward error recovery, (c) example of forward error recovery.