Deadlock prevention for flexible manufacturing system

Gang XU, Zhiming WU
(Automation Department, Shanghai Jiaotong University, Shanghai 200030, China)

Abstract: Deadlock must be prevented via the shop controller during the flexible manufacturing system (FMS) performing. Various models have been tried for the analysis and design of shop controller. Petri net is suitable to describe the dynamic behavior of the discrete event system, such as concurrency, conflict and deadlock, however, the verification of the system behavior needs structure analysis with complex theoretical proof method. Temporal logic model checking has important advantages over traditional theorem prover. It is fully automatic and can produce possible counter-example which is particularly important in finding subtle error in complex transition systems. In this paper, a new method for the deadlock prevention based on Petri net and Temporal Logic model checking is presented. The specification in the Temporal Logic is expressed according to some result of structure analysis of the Petri net. The model checking is employed to execute the formal verification, which will conduct an exhaustive exploration of all possible behaviors. Finally, an example is presented to demonstrate how the method works.

Keywords: Flexible manufacturing system (FMS); Scheduling; Deadlock

1 Introduction

Some resource allocations under performance may lead to deadlock situations. A deadlock is a state where a set of parts is in "circular waiting" i.e. each part in the set waits for a resource held by another part in the same set. Therefore, it is important to develop efficient algorithm to improve and optimize the system performances while preventing deadlock situations.

A very attractive and increasingly appealing alternative to simulation and testing is the approach of formal verification. While simulation and testing explore some of the possible behaviors and scenarios of the system, leaving open the question of whether the unexplored trajectories may contain the fatal bug, the formal verification conducts an exhaustive exploration of all possible behaviors. Thus, when a design is pronounced correct by a formal verification method, it implies that all behaviors have been explored, and the questions of adequate coverage or a missed behavior become irrelevant.

Model checking is one of the formal verification methods, a desired behavioral property of a system is verified over a given system (model) through exhaustive enumeration (explicit or implicit) of all the states reachable by the system and the behaviors that traverse through them. Compared to other approaches, model checking method enjoys two remarkable advantages, one is fully automatic and the other can supply counterexample when the design fails to satisfy a desired property. The advent of symbolic model checking revolutionized the field of formal verification and transformed it from a purely academic discipline into a viable practical technique.

In [1], a model checking approach is used in the synthesis of controller programs for a variety of robotics and manufacturing tasks. Hartonas et al. in [2,3] present new risk analysis tools based on model checking for the verification of industrial control designs. For the first time, these papers use symbolic model checking in the design of deadlock-free FMS. Because of lacking general description form of deadlock in FMS, the deadlock prevention in these papers is very simple. Petri nets are well-known approaches for specifying logic control system, and are suitable to describe the dynamic behavior of the discrete event system. Barkaoui et al. in [4,5] present a deadlock avoidance and prevention method for system of simple sequention processes with resources(S^3PR) based on structural theory of Petri nets. It is composed by two phases. The first phase is to add to the S^3PR net local control places such that all uncontrolled siphons of the S^3PR net become controlled. When deadlock-freeness of the augmented S^3PR net is not guaranteed, modify the initial markings of local control places. Thus, a controlled S^3PR is obtained. In [6,7], an
approach using signal interpreted petri net (SIPN) and model checking for the verification is presented. In this approach, the requirements asserted by users can be directly verified using TL formulas. Counterexamples serve also as a good debugging tool. J. Ezpeleta et al. in [8] has proposed a policy for resource allocation based on adding of new places to the net imposing restrictions that prevent the presence of unmarked siphons that may directly generate deadlocks. The crucial point of the procedure is the complexity that involves in computing the set of siphons of the Petri net model. For a general class of Petri net models, in [9] both deadlock prevention and avoidance control policies are proposed. The first part is based on the net reachability graph, while the second part is based on a look-ahead procedure that searches for deadlock situations by simulating the evolution process of system for a pre-established number of steps. Due to the fact that the avoidance policy does not assure that deadlocks are not reachable in future, they propose to combine this policy with a deadlock recovery system. In [10] a deadlock avoidance algorithm is proposed for a class of Petri net models formed for flow shop manufacturing where a set of sequential processes are executed without alternating the order of using resources in each case. The algorithm controls the input flow of new tokens in a local area, assuring that token evolutions in system are always possible. In [11] Wu et al. point out that, if an automated manufacturing system (AMS) operates at the deadlock boundary, i.e. under the maximally permissive control policy, it will not be deadlocked but a blocking may occur more likely. He presents an AMS that works near but not at the deadlock boundary in order to gain the highest productivity. For the first time he presents such a policy: L-policy. Without being too conservative, it can effectively reduce or even eliminate the blocking possibility that exists under a maximally permissive control policy. In [12], the definitions and criteria based on signal interpreted petri nets (SIPN) for formal correctness of logic controllers are presented, and the criterial can be checked automatically by computer algorithms.

In this paper, a new method based on Petri net and temporal logic model checking is presented. The specification in the temporal logic is expressed according to some result of structure analysis of the Petri net. No empty siphon in the system means no deadlock. The model checking is employed to execute the formal verification, which will conduct an exhaustive exploration of all possible behaviors.

This paper is organized as follows; the concept of siphon is presented in Section 2, model checking is shown in Section 3, and the algorithm is given in Section 4. In Section 5, an example is given to demonstrate the algorithm, Section 6 is a conclusion.

2 Basic structural results of Petri nets

Let $N = (P, T, F, W)$ be a Petri net where $P$ is a set of places, $T$ is a set of transitions, $F$ is a set of arcs and $W$ is the weight function for arcs.

Definition 1 An integer vector $f (f \neq 0)$ indexed by $P (f \in \mathbb{Z}^{|P|})$ is a p-invariant if it satisfies $f \cdot C = 0$ ($C$ is the incidence matrix and $0$ is a column vector where every component equals zero). If $f \in \mathbb{N}^{|P|}$, $f$ is called a p-semiflow. The positive support of $f$ is the set defined by: $\{p \in P \mid f(p) > 0\}$ and the negative support of $f$ is the set defined by: $\{p \in P \mid f(p) < 0\}$.

Definition 2 Let $D \subseteq P$, $D \neq \emptyset$.

1) $D$ is called a siphon if and only if $\forall D \subseteq D^+$, the siphon $D$ is minimal if and only if it includes no other siphon as a proper subset.

2) $D$ is called a trap if and only if $D \subseteq C \subseteq D$.

3) $D$ is said to be max marked at a marking $M$ if and only if $\exists p \in D$ such that $M(p) \geq \max\{W(p, t) \mid t \in P\}$.

Definition 3 Let $(N, M_0)$ be a marked Petri net. $(N, M_0)$ is said to be:

1) deadlockable if and only if $\exists M \in R(N, M_0)$, under which $\exists t \in T/M[t > M]$, is called a dead-marking, $R(N, M_0)$ (is the set of all reachable markings of $N$).

2) live under if and only if $\forall M \in R(N, M_0)$, and $\forall t \in T$, $\exists M' \in R(N, M): M'[t > .$

3) deadlock-free under $M_0$ if $\forall M \in R(N, M_0)$, $\exists t \in T/M[t > .$

Property 1 $N$ is said to be well-marked if all its siphons are marked at $M_0$. If $N$ is not well-marked, $N$ is obviously not live.

Property 2 Let $M$ be dead-marking of $N$, then $\exists$ a siphon $D$ token-free at $M$.

Definition 4 Controlled-Siphon: a siphon $D$ is controlled in $N$ if and only if $\forall M \in R(N, M_0)$, and $\forall t \in T/M[t > .$

Definition 5 Trap-controlled siphon: a siphon $D$ is