Expert Systems with Applications 38 (2011) 13555–13562

Contents lists available at ScienceDirect

Expert Systems with Applications

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

Behavioral modeling and verification of multi-agent systems for manufacturing control

W.L. Yeung

Department of Computing and Decision Sciences, Lingnan University, Hong Kong

ARTICLE INFO

Keywords:
Multi-agent systems
Deadlock
Formal methods
Model checking
Manufacturing control

ABSTRACT

Simulation is the primary approach to the performance analysis of multi-agent manufacturing systems. In a typical simulation experiment, varying the agent negotiation protocol's timing parameters can yield incomparable results in performance terms due to behavioral problems such as deadlocks and livelocks. Formal verification can help resolve such problems and therefore plays an important role in the design of agent negotiation protocols. We describe an efficient and scalable approach to formal verification based on an industrial-strength model checking tool and illustrate it with an example.

1. Introduction

Overcapacity, global market competition, short product life cycles are just some of the trends that put pressure on the manufacturing industry to become more agile and responsive to market changes (Schild & Bussmann, 2007). At the shop floor level, the conventional way to implement agile manufacturing systems with centralized control software is difficult (Parunak, 1996) and the potential single point of failure of the central controller poses a significant problem to the centralized approach. Multi-agent systems (MAS) provide a decentralized control architecture that can reduce complexity, increase flexibility, and enhance fault tolerance for manufacturing control (Krothapalli & Deshmukh, 1999). Other applications of MAS in manufacturing include supply chain management (Lee & Kim, 2008), workflow management (Yang, Sung, Wu, & Chen, 2010) and knowledge management (Wu, 2001).

A recent review of the literature has identified over a hundred applications of MAS in manufacturing systems and supply chain management (Lee & Kim, 2008). Many of these applications involve carrying out performance analysis case studies through simulation (e.g. Pendharkar, 1999; Wong, Leung, Mak, & Fung, 2006a). However, it has been argued that simulation is well suited for performance analysis of manufacturing systems but fails with respect to behavioral analysis, which is essential for addressing concurrency-related behavioral problems such as deadlocks and livelocks (Bos & Kleijn, 2002). For instance, a multi-agent manufacturing system is simulated with varying lengths of agent decision time. The aim of the experiment is to determine the average cycle time for a job but it turns out that not every simulation run completes successfully—some of the runs with particular parameter values actually stop abruptly in deadlocked states. Hence, the validity of simulation results is undermined by behavioral problems in the simulation model. Multi-agent manufacturing systems exploit distributed processing and are therefore particularly susceptible to these problems as discussed in (Parunak, 1987).

Formal methods have been suggested for the behavioral analysis of manufacturing systems (Bos & Kleijn, 2002). Formal methods are mathematical techniques that have successfully been applied to the specification and analysis of various types of industrial systems (Abrial, 2006). While traditional formal methods are often considered as laborious and have limited scalability in practical applications, recent developments in model checking (Clarke, Grumberg, & Peled, 1999) have vastly increased the viability and scalability of formal methods for industrial applications. Model checking involves specifying a formal behavioral model of the system design and verifying certain behavioral properties (e.g. absence of deadlock) of the model by exploring the model's state space completely. Bos and Kleijn (2002) have shown how model checking can be integrated with simulation for analyzing both performance and behavioral properties of manufacturing systems. However, their case studies do not address agent-based control.

Critical to the proper functioning of a multi-agent control architecture is the design of agent negotiation protocols (Krothapalli & Deshmukh, 1999). Previous attempts (e.g. Billington, Gupta, & Gallasch, 2008; Dang & Huhns, 2006) to verify negotiation protocols did not address adequately the *timing* aspect of negotiation processes. In this paper, we extend the use of model checking to the design of negotiation protocols for multi-agent manufacturing systems. Our method involves the formalism of Communicating Sequential Processes (CSP) (Hoare, 1985) and the FDR model checking tool (Formal Systems (Europe). CSP is a process algebra for the analysis and reasoning of concurrency and communication

E-mail address: wlyeung@ln.edu.hk

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.04.067

in general. In particular, it supports the reasoning of nondeterminism and time. CSP has been applied to the verification of communication and security protocols (Ryan, Schneider, Goldsmith, Lowe, & Roscoe, 2001) as well as various types of concurrent/distributed systems including real-time and fault-tolerant systems (He & Hoare, 1987; Yeung & Schneider, 2003, 2005).

Our work contributes to the design of simulation experiments for production systems as variation of *timing* parameters in a simulation experiment can yield different results in not only performance, but also behavioral terms.

The next section reviews the contract net approach to agent-based manufacturing control and elaborates on the rationale for behavioral analysis. Section 3 briefly introduces some essential concepts and notations of the formalism of CSP. Section 4 presents an example in the formal verification of negotiation protocols for multi-agent manufacturing systems. Section 5 discusses the merits of our approach with reference to other related approaches. Section 6 gives a conclusion.

2. Background and rationale

2.1. The control net approach to manufacturing control

An agent can be defined as a computer system that is situated in some environment and that is capable of autonomous actions in this environment in order to meet its design objective (Woolridge, 2001). A multi-agent system can be considered as consisting of a number of agents which reason autonomously, interact with one another, adapt to environmental changes, and pursue advantages in a rational manner (Lee & Kim, 2008). Critical to the functioning of a multi-agent system is the design of a agent negotiation protocol which governs the interactions among agents.

A prominent approach to designing such protocols for industrial applications is based on the Contract Net protocol (CNP) (Foundation for Intelligent Physical Agents, 2002; Smith, 1980) which was originally proposed for cooperative problem solving in a distributed processing environment based on a negotiation metaphor (Davis & Smith, 1983). CNP-based multi-agent systems have traditionally been applied in the manufacturing industry for manufacturing control (Duffie & Piper, 1986; Parunak, 1987), process planning and scheduling (Gu, Balasubramanian, & Norrie, 1997; Lim & Zhang, 2004), production capacity allocation (Brandolese, Brun, & Portioli-Staudacher, 2000), vehicle routing (Sandholm, 1993), and supply chain management (Lu & Wang, 2008). Other applications in manufacturing include computer-supported collaborative work (Lemaitre & Excelente, 1998).

In a typical application of multi-agent systems in manufacturing control based on CNP (e.g. Gu et al., 1997; Krothapalli & Deshmukh, 1999; Lim & Zhang, 2004; Wong et al., 2006a, Wong, Leung, Mak, & Fung, 2006b), the allocation of parts on machines is accomplished through a process of interaction between agents representing parts and agents representing machines. The basic principles of this approach can be described as follows. A part agent announces the processing requirements of a part to all machine agents in a task announcement message. Such a message is checked by every machine agent against its own capabilities and capacity to decide whether to respond with a bid message. Bid messages received by the part agent are evaluated and an award message is sent to the selected bidding (machine) agent.

Researchers have applied the contract bidding paradigm and the associated design principles in a number of MAS applications in manufacturing control with various physical settings. In some cases (e.g. Krothapalli & Deshmukh, 1999; Kumar, Tiwari, & Chan, 2008; Wong et al., 2006a), they employed two or more schemes on which agents' decisions were based and the different schemes were compared based on the results of simulation experiments conducted under the same physical and workload settings. The comparison addressed performance measures such as average waiting time of parts and machine utilization.

2.2. Dealing with agent negotiation overheads

A CNP-based negotiation process involves the transmission and processing of messages by agents as overheads in terms of processor time and network traffic which can ultimately affect system performance. This issue, known as the *message congestion problem*, was raised in (Smith, 1980) and followed up in (Sandholm, 1993; Sandholm & Lesser, 1995). Gu et al. (1997) & Krothapalli & Deshmukh (1999) highlighted this issue in the context of multi-agent manufacturing systems.

As discussed in (Smith, 1980), there can be tradeoffs in dealing with these overheads in the detailed design of the negotiation protocol. For instance, a bidder can be programmed to wait for the result of a bid for only a certain amount of time before assuming that the bid has failed; to reduce their waiting time, contractors can be programmed to announce results to bidders at the expense of increased message traffic.

Tilley (1996) carried out a detailed simulation study on the impact of two major timing parameters, namely task announcement time and task evaluation time, on the performance of a CNP-based multi-agent manufacturing system. The results show that the latter parameter can significantly influence the efficiency of the agent negotiation process and hence the overall system performance.

The above suggests that there is often the need to fine-tune the logic and timing parameters in the detailed design of a negotiation protocol for performance purposes. In doing so, care must also be taken against potential behavioral problems such as deadlocks and livelocks.

3. Communicating sequential processes

In this section, we describe some basic concepts and notations of Communicating Sequential Processes (CSP) (Hoare, 1985) used in the examples in Section 4.

In the language of CSP, a process is described in terms of the possible interactions it can have with its environment, which may be thought of as another process or set of processes. Interactions are described in terms of instantaneous atomic synchronisations, or *events*. A process can be considered as a "black box" with an interface containing a number of events through which it interacts with other processes. The set of all events in the interface of a process P, written αP, is called its *alphabet*. It is important to note that interface events are intended as synchronisations between the participating processes and not as autonomous actions under the control of a single process.

The following paragraphs briefly introduce the CSP operators used in this paper. A comprehensive description of the language is found in (Hoare, 1985; Roscoe, 1998). The language of CSP used in this paper is defined by the following pseudo Backus-Naur form definition:

$$\begin{aligned} \mathrm{P} ::= \; & Stop \mid a \rightarrow \mathrm{P} \mid a : A \rightarrow P_a \mid \mathrm{P} \Box \mathrm{P} \mid \mathrm{P} \sqcap \mathrm{P} \mid \\ & \mathrm{P} \triangleright \mathrm{P} \mid \mathrm{P} \underset{A}{\parallel} \mathrm{P} \mid \mathrm{P} \setminus A \end{aligned}$$

where Σ is the set of all possible events, a ranges over Σ, and $A \subseteq \Sigma$.

Let a and b be events and P, Q, and R be CSP processes. The process $Stop$ is the deadlocked process, unable to engage in any events or make any progress. The prefix process $a \rightarrow P$ is ready to engage