Journal of Manufacturing Systems xxx (2013) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Journal of Manufacturing Systems

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

Technical Paper

A bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling

Yu-Cheng Chou [a,*], Huajun Cao [b], Harry H. Cheng [c]

[a] Department of Mechanical Engineering, Research and Development Center for Microsystem Reliability, Institute of Biomedical Technology, Chung Yuan Christian University, Zhongli, Taoyuan 32023, Taiwan
[b] Institute of Manufacturing Engineering, Chongqing University, Chongqing 400044, China
[c] Department of Mechanical and Aerospace Engineering, University of California at Davis, Davis, CA 95616, USA

ARTICLE INFO

Article history:
Received 11 December 2011
Received in revised form 13 January 2013
Accepted 22 January 2013
Available online xxx

Keywords:
Autonomic system
Mobile agent-based system
Distributed
Manufacturing
Flexible job shop scheduling
Dynamic job shop scheduling
Autonomic computing

ABSTRACT

This paper presents a bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling. The system matches the autonomic system architecture, inspired by the autonomic nervous system and proposed by the IBM, and has the IBM-defined fundamental self-managing properties, so that it can manage itself with little human intervention. The system conforms to the IEEE FIPA (Foundation for Intelligent Physical Agents) standard. Therefore, the interoperability between agents of the system and agents from many active heterogeneous FIPA compliant agent platforms can be ensured. The system supports the execution of C/C++ mobile agent codes. Thus, it is applicable to a variety of applications, especially for distributed mechatronic and embedded systems. In addition, since the system is composed of agents, including stationary and mobile agents, the system has a high scalability and flexibility to integrate and adopt various scheduling models and algorithms for different scheduling requirements. An overall architecture of the system and critical implementation details about the agency and agents in the system are presented in this article. An energy saving job shop scheduling example is used to validate one autonomic property of the system.

1. Introduction

As the size of manufacturing systems rapidly increases, it has been the trend to adopt decentralized and distributed techniques to control and manage those systems in order to reduce complexity and cost, increase flexibility, and enhance fault tolerance. Meanwhile, most manufacturing systems, such as job shops, operate in a dynamic environment where inevitable and unpredictable disturbances necessitate the revision of established schedules during manufacturing processes.

An agent is an encapsulated computer system that can autonomously perform reactive, proactive, and social actions in its execution environment [1]. In the area of design and manufacturing, a manufacturing resource, such as a machine or an operator, may cooperate and negotiate with other agents for task assignment; an existing engineering software may be integrated with a distributed design and manufacturing system. As opposed to traditional systems, agent-based systems do not have a centralized system control structure and pre-defined agenda for the system execution [2]. Thus, agent technology can significantly enhance the design and analysis of systems whose problem domain is distributed, and whose subsystems exist in a dynamic environment and need to interact with each other more flexibly [3].

A comparative study showed that multi-agent approach is a promising technique in dynamic manufacturing scheduling due to its autonomy, flexibility, modularity, robustness, and heterogeneity [4]. In addition, agent-based approaches have the following advantages over traditional approaches for distributed manufacturing scheduling [5,6].

(1) They facilitate the building of robust and efficient scheduling systems, because they employ parallel computation through a large number of processors.
(2) They facilitate the integration of manufacturing process planning and scheduling.
(3) They lead to cooperative scheduling, because they allow individual resources to trade off local performance to improve global performance.
(4) They facilitate the building of reliable and fault-tolerant scheduling systems, because agents can directly connect to physical devices in order to realize real-time dynamic rescheduling.

Abbreviations: SCY, scheduling center agency; SCT, scheduling center agent; MCY, manufacturing cell agency; MCT, manufacturing cell agent; BT, bidder agent; OT, order agent.

* Corresponding author. Tel.: +886 3 2654316.
E-mail addresses: ycchou@cycu.edu.tw (Y.-C. Chou), hjcao@cqu.edu.cn (H. Cao), hhcheng@ucdavis.edu (H.H. Cheng).

0278-6125/$ – see front matter © 2013 The Society of Manufacturing Engineers. Published by Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.jmsy.2013.01.005

(5) Schedules are obtained by negotiation rather than search. Thus, the manufacturing capabilities of manufacturers can be shared with each other, and optimization is allowed across all levels, namely, the supply chain level, shop floor level and enterprise level.
(6) Different techniques, such as simulated annealing [7] and genetic algorithms [8,9], can be incorporated at different levels for decision-making purposes.

Listed below introduces some multi-agent based approaches employed in dynamic manufacturing scheduling. A just-in-time dynamic scheduling approach was proposed in [10], where jobs and machines were modeled as agents that negotiate using market-based contract net protocol (CNP). A dynamic scheduling approach based on mediator and contract net protocol was presented in [11]. The approach models parts and machines as agents, and machine mediator and resource mediator as manager and coordinator of agents, respectively. A response threshold model for dynamic scheduling with flexible routing and sequence dependent setups was introduced in [12], where only machines were modeled as agents. An Ant Colony Intelligence model was presented in [13], where both machine selection and job sequencing problems were considered. A hierarchical control architecture with five CNP rules for flexible manufacturing system (FMS) scheduling was proposed in [14], where both job selection and machine selection rules were considered. A multi-agent system (MAS), called MASDSche-GATS, for distributed manufacturing scheduling with genetic algorithm and Tabu search was introduced in [15,16], where agents were able to find the local solution by genetic algorithm or Tabu search, and cooperated with one another to achieve global schedule. A MAS approach that modeled both tasks and resources as agents was presented in [17], where ant colony optimization was adopted to schedule both original and dynamic tasks. A multi-agent scheduling system was developed in [18] to solve job shop scheduling problems that consider dynamic events as well as routing and process flexibility. In [19], an agent-based parallel genetic algorithm for job shop scheduling problems was proposed, which improved the performance and quality of solutions for a genetic algorithm. A multi-agent Tabu search model, called FJS MATSLO+, was introduced in [20], where local optimization and new diversification techniques were employed to solve flexible job shop problems. In [21], a parallel modular simulated annealing algorithm was developed to tackle classical job shop problems through multi-agent systems. In [22], an agent-based service-oriented integration architecture was proposed to leverage manufacturing scheduling services on a network of virtual enterprises.

Many multi-agent systems applied to manufacturing are based on stationary agents. Mobile agents have all the features of stationary agents with an additional mobility attribute that creates the following advantages over stationary agents:

(1) *Local interaction*: A mobile agent can go to a remote machine to interact locally with other agents residing in that machine, thereby reducing the network load.
(2) *Parallel execution*: Multiple mobile agents can go to different remote machines to perform same or different tasks in parallel, thereby enhancing the efficiency.
(3) *Disconnected operation*: A mobile agent can operate without an active connection between itself and the machine where it is created.

An investigation showed that mobile agent technology can improve system integration and agility in the distributed manufacturing domain [23]. Several mobile agent-based manufacturing systems or mechanisms were developed in the past. A multi-agent protocol, ECNPro (the Extended Contract-Net-like multilateral Protocol) [24], was developed for handling buyer–seller negotiations in supply chain management. A software architecture, A^{3}M [25], was proposed to handle, within a single manufacturing cell, automatic assignment of control tasks to controllers, monitoring of cell functionalities, and dynamic cell reconfiguration. An integrated hierarchical framework based on mobile agent technology was proposed as an approach to resolve problems of scalability and management efficiency in large-scale networked manufacturing domains [26]. A mobile agent-based manufacturing decision system was prototyped to demonstrate the ability of mobile agents to support interoperable STEP-NC compliant manufacturing [27]. A flexible maintenance system integrating mobile agent technology and wireless sensor network was developed and applied to a real-world numerical control machining center [28]. A manufacturing control system was designed as a self-organizing multi-agent system where three main kinds of mobile agents were employed for coordination purposes [29]. A negotiation mechanism, MAN-Pro (Mobile Agent-based Negotiation Process) [30], was proposed to handle the information exchange in a shop floor control system. A two-level scheduling model was implemented using mobile agents to handle high-level process scheduling and low-level node scheduling for manufacturing chains [31].

This paper presents an autonomic mobile agent-based system for distributed job shop scheduling. The main features of the presented system, that are different from those of aforementioned systems or mechanisms, include:

1. The system matches the autonomic system reference architecture, which was inspired by the human body's autonomic nervous system and proposed by the IBM. Thus, the system has the four IBM-defined fundamental self-managing properties, namely, self-configuration, self-optimization, self-healing, and self-protection [32,33]. This system can manage itself with little human intervention.
2. The system conforms to the IEEE FIPA (Foundation for Intelligent Physical Agents) standard [34]. This compliance ensures the interoperability between its agents and other agents from an increasing number of FIPA compliant, heterogeneous agent platforms.
3. The system supports the execution of C/C++ mobile agent codes. This feature enables the system to be applicable to a wide range of applications due to C/C++'s comprehensive functionality, broad use, and international standard.

The rest of the article is organized as follows. Section 2 illustrates the system architecture. Section 3 presents the system implementation. Section 4 validates the presented system's self-configuration property using an energy saving job shop scheduling example. Section 5 draws the conclusion of this article.

2. System architecture

This section illustrates the architecture of the prototype autonomic mobile agent-based system. As shown in Fig. 1, the system contains two kinds of agencies, scheduling center agency (SCY) and manufacturing cell agency (MCY). An agency in this article is an agent platform where agents operate. In this prototype system, there is one scheduling center that deals with job shop scheduling in multiple manufacturing cells. Thus, there are one SCY and multiple MCYs, MCY_1 to MCY_N, in Fig. 1. The scheduling center agent (SCT) and manufacturing cell agent (MCT) are stationary agents. They are not dynamically generated and do not travel to other agencies.

When the SCT receives a new order from the order database, it creates as many bidder agents (BTs) as MCTs. A BT is one kind of

Please cite this article in press as: Chou Y-C, et al. A bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling. J Manuf Syst (2013), http://dx.doi.org/10.1016/j.jmsy.2013.01.005

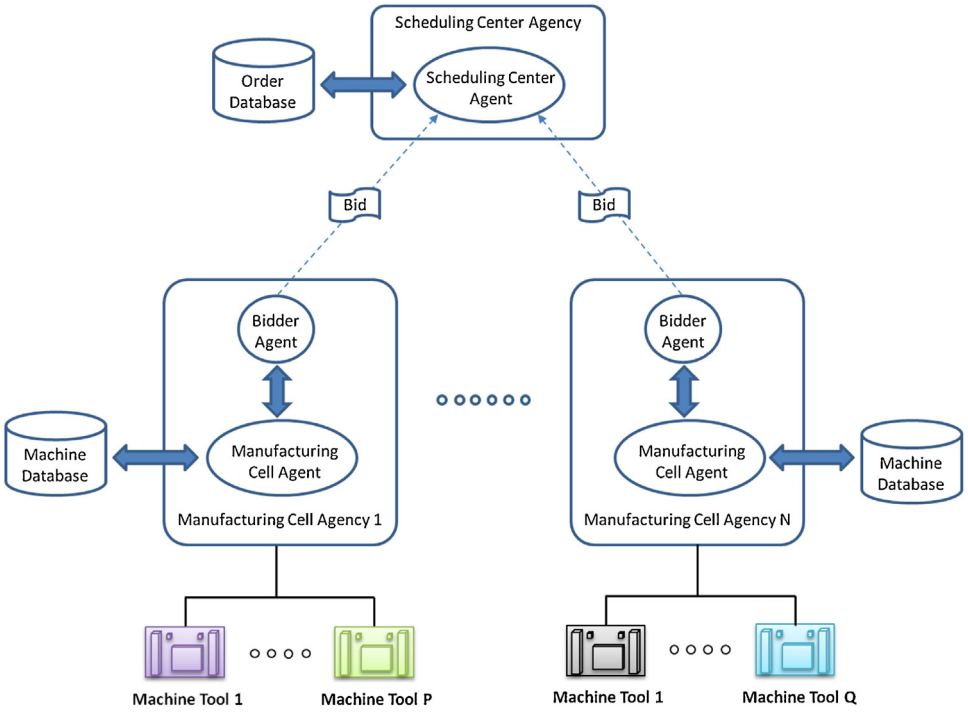

Fig. 1. The scheduling center agent creates multiple bidder agents that migrate to their corresponding manufacturing cell agency to generate a bid by interacting with the local manufacturing cell agent. Each bidder agent informs the scheduling center agent of the generated bid through an FIPA Agent Communication Language message.

mobile agent in this system. Each BT travels to the corresponding MCY to interact locally with the MCT to create a bid. Then each BT informs the SCT of the generated bid through a FIPA Agent Communication Language (ACL) message, and destroys itself afterward. A bid is created based on the information obtained from the machine database through an MCT. A bid is an index that represents the capability of a manufacturing cell to carry out an order.

The SCT selects the best bid among all the received bids and creates another mobile agent, order agent (OT). The OT will travel to the corresponding MCY, MCY_K, to perform its tasks by interacting with the local MCT and, if needed, with the SCT, as shown in Fig. 2. After its tasks are all done, the OT will destroy itself.

In this article, each of the four autonomic properties is defined as follows.

2.1. Self-configuration

The system can identify the most appropriate manufacturing cell to carry out a new order according to the capability index, and generate an initial optimal schedule for the selected manufacturing cell according to the optimization objective.

2.2. Self-optimization

The system can optimize its overall performance through re-scheduling every time when the performance index drops to a certain extent.

2.3. Self-healing

The system can recover from faults through re-scheduling every time when a serious event, such as the machine breakdown, occurs during operations.

2.4. Self-protection

The system can prevent faults from happening, such as the machine breakdown and overloading, through re-scheduling every time when a machine's status factor deviates beyond a threshold.

The general and overall process of creating the proposed agent-based autonomic system includes the creation of four different agents, namely, the SCT, BT, OT, and MCT, which play different roles in the autonomic system. The self-configuration manager of the system consists of the SCT, BTs, and OT, as shown in Fig. 3. Each BT performs the monitoring function of the manager. The SCT performs the analysis and planning functions of the manager, whereas the OT performs the execution function of the manager. In addition, the OT also plays the roles of self-optimization, self-healing, and self-protection managers, as depicted in Fig. 4. Therefore, the OT integrates the monitoring, analysis, planning, and execution functions to maintain the system's self-optimization, self-healing, and self-protection properties. As shown in Fig. 5, each MCT plays the role of manageability endpoint, which performs functions of sensors and effectors. As illustrated in Fig. 6, all the machine tools within a manufacturing cell are considered a managed element as a whole. In this system, an autonomic manager communicates with a manageability endpoint to obtain the desired information of machine tools that operate in a manufacturing cell.

The critical challenges involved in building the proposed autonomic system include the development and implementation of various algorithms in order to realize the self-configuration, self-optimization, self-healing, and self-protection properties of the system for practical manufacturing scheduling purposes.

In this paper, self-optimization, self-healing, and self-protection processes are all performed within a manufacturing cell, which indicates that a selected manufacturing cell can still complete an order through re-scheduling. In addition, if an event causes a selected manufacturing cell to be incapable of completing an order, the OT on that manufacturing cell will inform the SCT about this situation. The SCT will then re-initiate the self-configuration process

Please cite this article in press as: Chou Y-C, et al. A bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling. J Manuf Syst (2013), http://dx.doi.org/10.1016/j.jmsy.2013.01.005

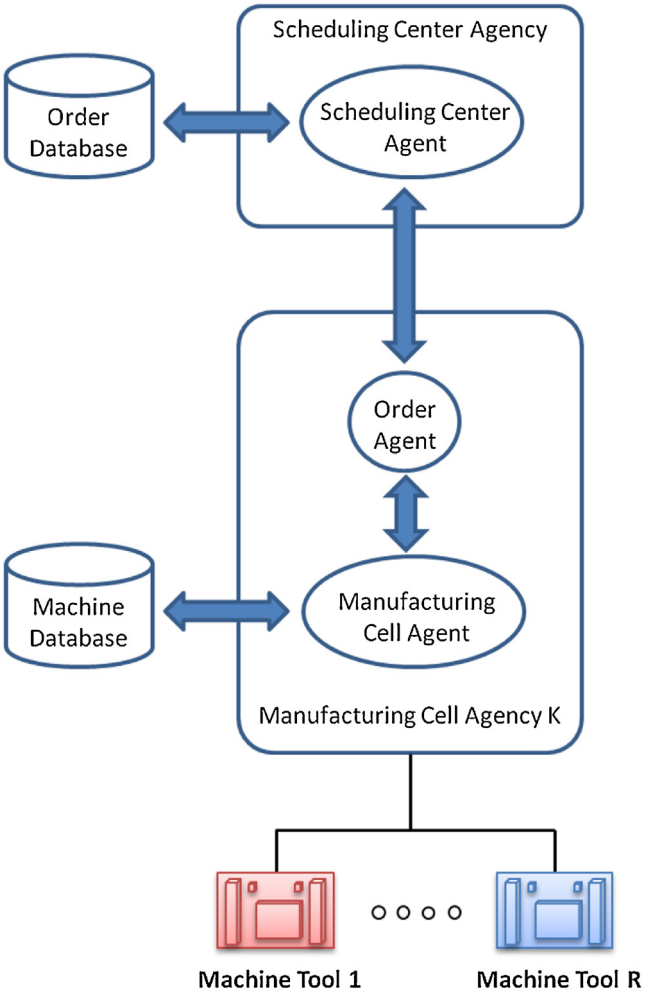

Fig. 2. According to the received bids, the scheduling center agent selects the most appropriate manufacturing cell and generates an order agent. The order agent then migrate to the corresponding manufacturing cell agency to perform its tasks by interacting with the local manufacturing cell agent and, if needed, with the scheduling center agent as well.

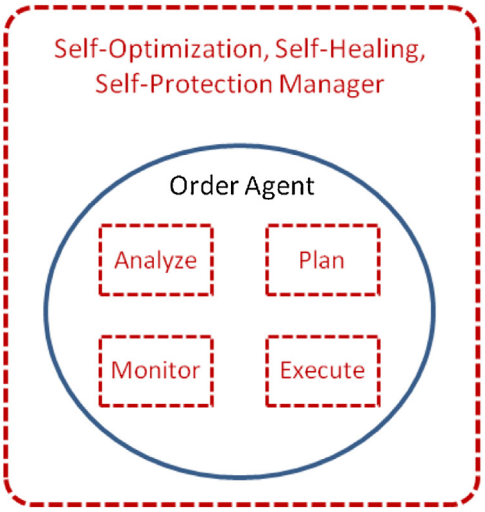

Fig. 4. Order agent plays the self-optimization, self-healing, and self-protection manager of the system.

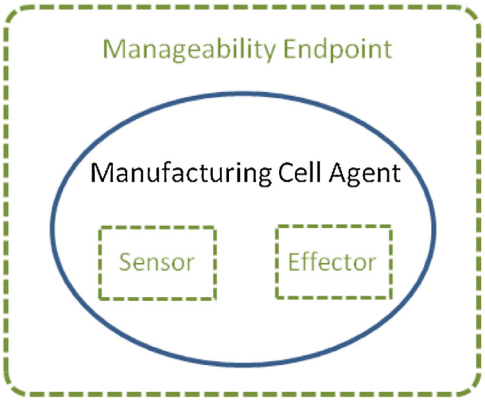

Fig. 5. Manufacturing cell agent plays the manageability endpoint of the system.

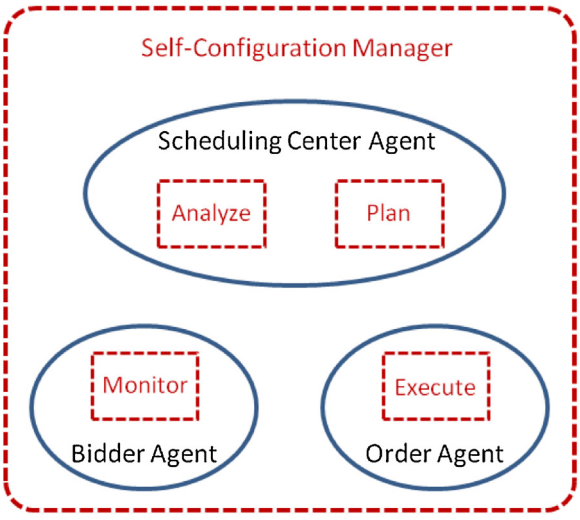

Fig. 3. Scheduling center agent, bidder agent, and order agent comprise the self-configuration manager of the system.

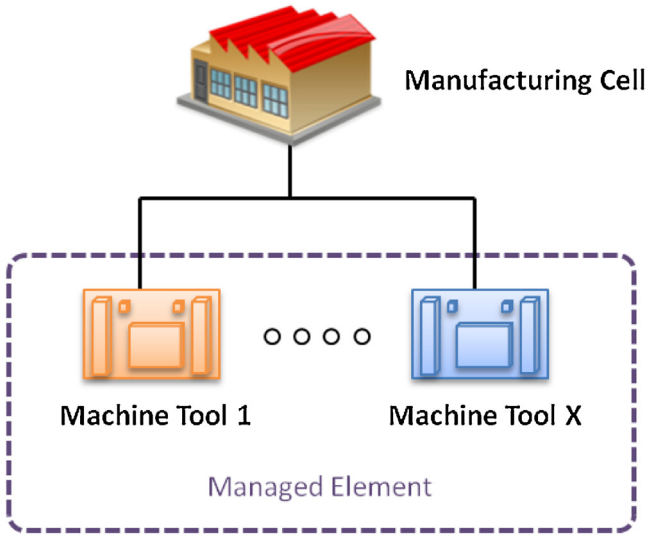

Fig. 6. Machine tools within a manufacturing cell are considered a managed element of the system.

Please cite this article in press as: Chou Y-C, et al. A bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling. J Manuf Syst (2013), http://dx.doi.org/10.1016/j.jmsy.2013.01.005

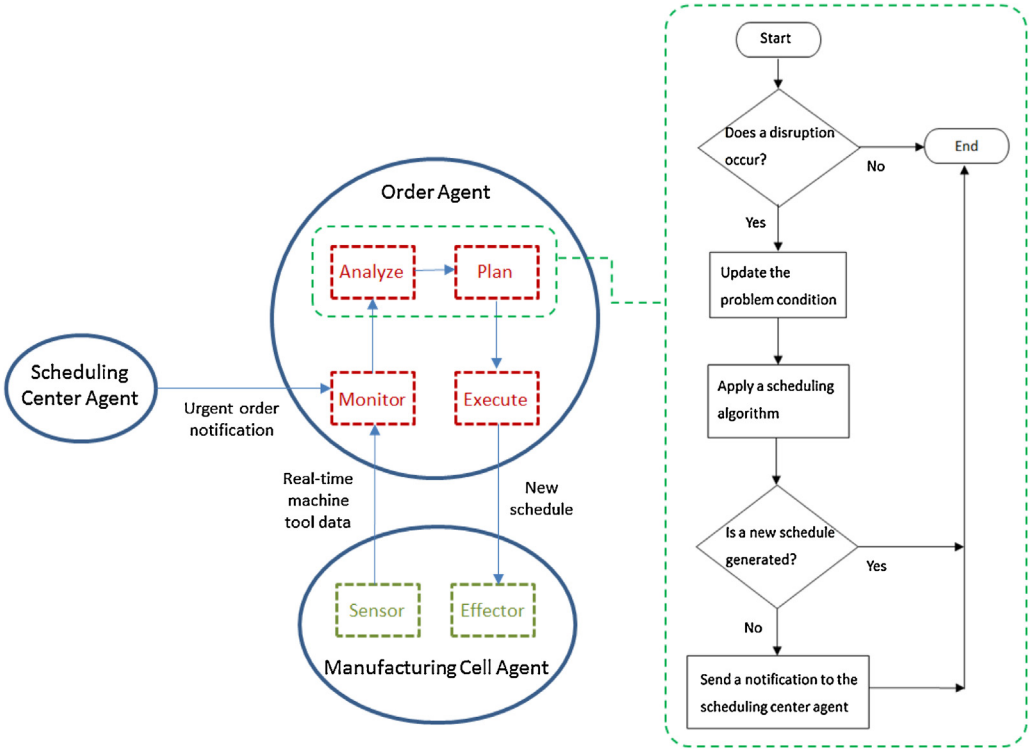

Fig. 7. The scheduling center agent, an order agent, and a manufacturing cell agent comprise the re-scheduling mechanism.

to identify a new manufacturing cell that is most appropriate to carry out the rest of the order.

The re-scheduling mechanism consists of three kinds of agents, namely, the scheduling center agent, an order agent, and a manufacturing cell agent, as shown in Fig. 7. As the self-optimization, self-protection, and self-healing manager, an order agent performs monitoring, analysis, planning, and execution functions. In addition, as the manageability endpoint, a manufacturing cell agent serves as a sensor and an effector. An order agent, through its monitoring function, receives real-time machine tools' data from a manufacturing cell agent and/or urgent order notifications from the scheduling center agent. The analysis and planning functions of an order agent perform the re-scheduling process, as shown in the flowchart of Fig. 7. In the re-scheduling process, the received information is evaluated to determine whether a disruption occurs. A disruption, or a re-scheduling factor, includes undesired events such as job delay, urgent order, resource unavailability, and missing tool/fixture. Once a disruption happens, the problem condition, meaning the requirements and constraints of the problem, will be updated, and a scheduling algorithm will be applied to create a new schedule. If a new schedule is obtained, it will be sent to a manufacturing cell agent through the execution function of an order agent. If a new schedule cannot be generated, meaning that such a disruption makes a manufacturing cell unable to complete the original order under the desired condition, a notification will be sent to the scheduling center agent, so that the scheduling center agent will perform self-configuration process to identify another manufacturing cell to carry out the rest of the order.

3. System implementation

This section presents some implementation aspects of the prototype system including the mobile agent platform, or agency, and each agent in the system.

3.1. Agency

Mobile-C [35–37], an embeddable mobile agent platform supporting C/C++ mobile agents for various distributed applications, is employed in the system as the scheduling center agency (SCY) and manufacturing cell agency (MCY), as shown in Fig. 8. Among the Mobile-C modules shown in Fig. 8, by default, the three FIPA mandatory modules, the Agent Management System (AMS), Agent Communication Channel (ACC), and Directory Facilitator (DF), are initialized when Mobile-C is started. The AMS is related to the creation, registration, execution, migration, persistence, and termination of a mobile agent. The ACC is related to the inter-agency mobile agent transport and inter-agent communication. The DF is related to yellow page activities. The Agent Security Manager (ASM) is responsible for maintaining the security policies for an agency. In this article, mobile agents are assumed authorized mobile agents. Thus, the ASM is not required to be initialized in Mobile-C. By default, the ASM is not initialized when Mobile-C is started.

In this article, a stationary agent, the scheduling center agent (SCT) or manufacturing center agent (MCT), has the same XML structure as a mobile agent, the bidder agent (BT) or order agent (OT). A C/C++ program source code is embedded as the agent code in this XML structure [38]. When the SCY or MCY is initiated on a computer, the SCT or MCT is loaded in the agency at runtime, and an Agent Execution Engine (AEE) is created to run the agent code, as shown in Fig. 8. Ch [39–41], an embeddable C/C++ interpreter suitable for computer-aided design and manufacturing applications, was adopted as the AEE in Mobile-C. When a mobile agent, the BT or OT, arrives at an MCY, a new AEE is automatically created to run the mobile agent code, as shown in Fig. 8.

3.2. Agents

This subsection presents a simplified flowchart of the agent code for each of the four agents in the system, namely, the scheduling

Please cite this article in press as: Chou Y-C, et al. A bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling. J Manuf Syst (2013), http://dx.doi.org/10.1016/j.jmsy.2013.01.005

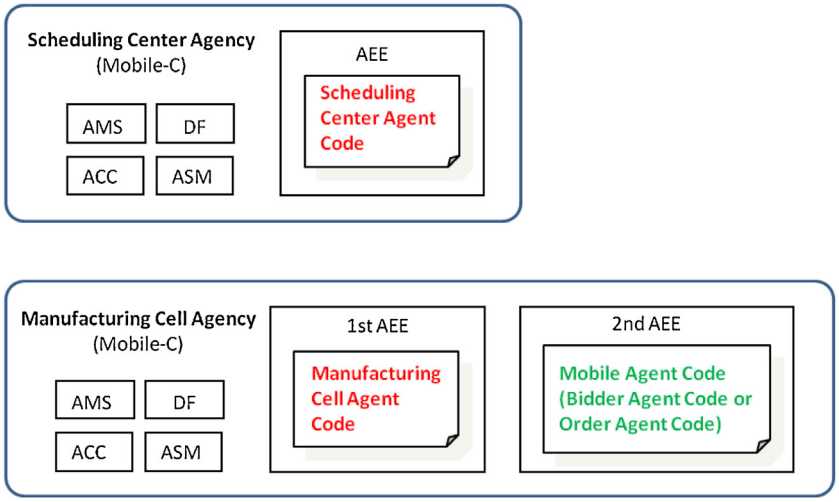

Fig. 8. Scheduling center agency and manufacturing cell agency of the system.

center agent (SCT), manufacturing cell agent (MCT), bidder agent (BT), and order agent (OT).

Fig. 9 shows a simplified flowchart for the scheduling center agent code. The SCT first checks the order database to see if there is a new order. When there is a new order, the SCT creates as many BTs as manufacturing cells. It then checks its mailbox to see if there is an ACL message sent by a BT. The message contains a bid generated by a BT at a remote manufacturing cell. The SCT also checks whether the ACL messages from all the BTs have been received. Once all the messages have been received, the SCT will create an OT that will migrate to the manufacturing cell offering the best bid.

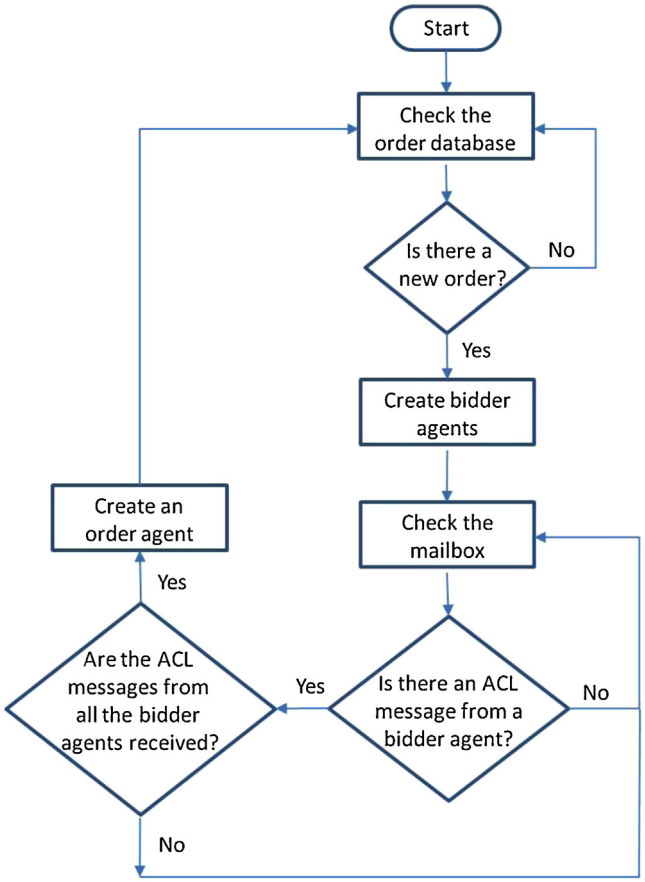

Fig. 9. A simplified flowchart for the scheduling center agent code.

Fig. 10 shows a simplified flowchart for the manufacturing cell agent code. The task of an MCT is simple. It continuously checks its mailbox to see if there is an ACL message. When there is a message, the MCT will perform the operations specified in the message. The operations may include obtaining specific information from the machine database, sending ACL messages that contain desired information to the BT or OT, writing specific information into the machine database, and so on. Having an MCT, rather than a BT or OT, access or manipulate the data relevant to machine operations makes the system more secure against faulty operations that could cause malfunction of machines.

Fig. 11 shows a simplified flowchart for the bidder agent code. After a BT is created at the SCY, it migrates to its destination MCY. The BT then sends an ACL message to the local MCT. The message specifies the information which the BT requests the MCT to obtain in order to generate a bid. Once the BT receives an ACL message that contains the desired information from the local MCT, it will create a bid based on a bid-generation algorithm, and then inform the SCT about the bid through an ACL message.

Fig. 12 shows a simplified flowchart for the order agent code. After an OT is created at the SCY, it migrates to its destination MCY. The OT then sends an ACL message to the local MCT. The

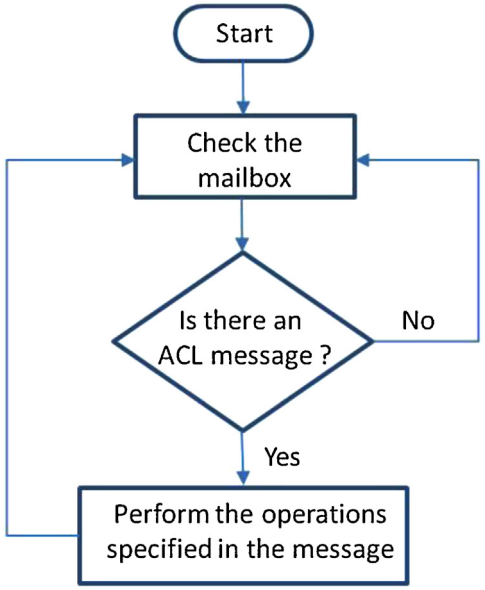

Fig. 10. A simplified flowchart for the manufacturing cell agent code.

Please cite this article in press as: Chou Y-C, et al. A bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling. J Manuf Syst (2013), http://dx.doi.org/10.1016/j.jmsy.2013.01.005

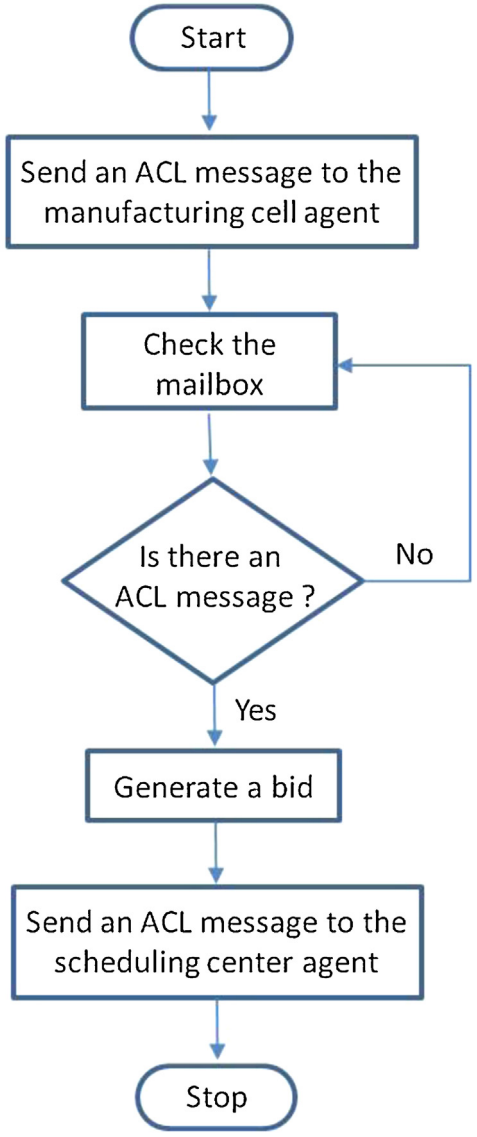

Fig. 11. A simplified flowchart for the bidder agent code.

message specifies the information which the OT requests the MCT to obtain for scheduling purposes. Once the OT receives an ACL message that contains the desired information from the local MCT, it will determine whether a scheduling operation is needed. If such an operation is needed, the OT will find an optimal schedule based on a default scheduling technique, such as a genetic algorithm, simulated annealing, and Tabu search. Afterward, the OT will send another ACL message to the MCT to inform the optimal schedule as well as request the desired information for the next decision-making cycle. On the other hand, if a scheduling operation is unnecessary, the OT will determine whether the order is completed based on the information from the message. When the order is found to be completed, the OT's task is finished as well. Otherwise, the OT will send another ACL message to the MCT to request the desired information for the next decision-making cycle.

In this article, an OT represents a scheduling technique, i.e. an algorithm, which obtains an optimal schedule with respect to a desired objective for a selected manufacturing cell. Once an algorithm or desired objective needs to be changed, which is decided by an OT or human operator, a new OT can be dynamically created at the SCY and migrate to a specified manufacturing cell to replace the OT on duty. This feature shows that various scheduling techniques

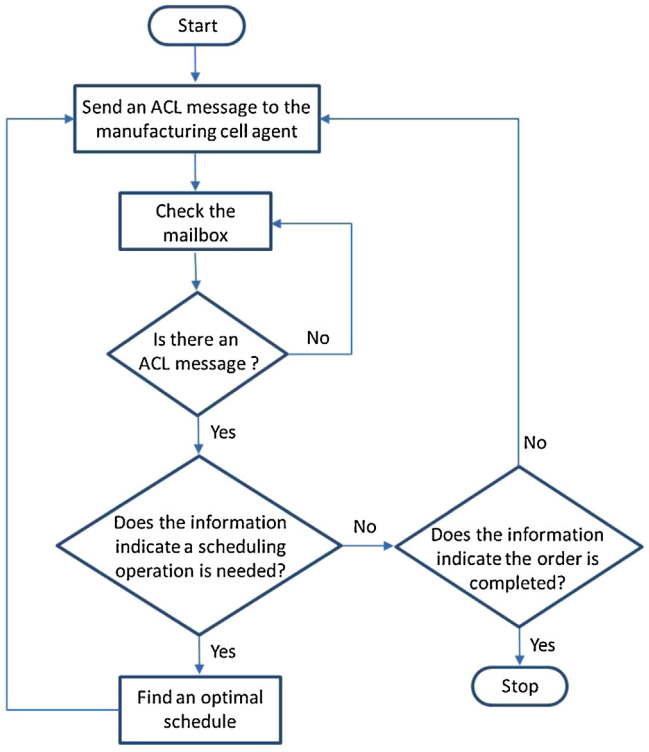

Fig. 12. A simplified flowchart for the order agent code.

can be dynamically incorporated into the system, thus indicating a high scalability and flexibility of the system.

4. An energy saving job shop scheduling example

This section uses an energy saving job shop scheduling example to validate the prototype system's self-configuration property.

Manufacturing is one of the major industries that consumes a lot of energy and pollutes the environment. Such a situation is more prominent among the manufacturing powers in the world. For instance, according to statistics, the total energy consumption of China's manufacturing industry accounts for 63% of China's national energy consumption [42]. Thus, reducing the energy use in manufacturing plays a key role in minimizing production costs as well as in realizing green manufacturing objectives. In addition, the energy consumed by a machine tool dominates the energy requirements for a manufacturing process on that machine tool and can significantly affect the lifespan, malfunction possibility, and damage possibility of that machine tool [43]. Moreover, because different machine tools usually have different powers and energy efficiencies, they consume different amount of energy to process even the same work pieces. Such differences in the energy consumption provide an opportunity for energy saving by optimizing the scheduling between work pieces and machine tools. Therefore, an optimal or near-optimal arrangement of workpieces on each machine in a manufacturing cell is an effective way to keep the total energy consumption down to the minimum, thereby reducing the cost on the manufacturing, maintenance, and repair activities.

An energy saving scheduling problem is typically described as: In a job shop, m machine tools are ready for use and n workpieces are to be processed. Each workpiece can have a different quantity. The problem is how to obtain the optimal scheduling solution, which produces the minimal energy consumption and satisfies constraints in the makespan and workpiece quantity, by dividing the workpieces into proper units, and assigning and sequencing them to the right machine tools. Such an energy saving scheduling

Please cite this article in press as: Chou Y-C, et al. A bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling. J Manuf Syst (2013), http://dx.doi.org/10.1016/j.jmsy.2013.01.005

problem is an NP problem. A mathematical model was developed to describe the problem and a genetic algorithm-based heuristic method was designed to obtain the optimal solution [44]. The mathematical model and heuristic method are summarized in the following subsections.

4.1. Mathematical model

Matrices and vectors used in the mathematical model are defined as follows.

$$\text{Quantity of each workpiece}: N = [n_1, n_2, \ldots, n_n] \quad (1)$$

$$\text{Processing time for one unit of each workpiece}: T = [t_1, t_2, \ldots, t_n] \quad (2)$$

$$\text{Minimum distribution quantity of each workpiece}: D = [d_1, d_2, \ldots, d_n] \quad (3)$$

$$\text{Available time of each machine}: T_a = [t_a^1, t_a^2, \ldots, t_a^m] \quad (4)$$

$$\text{Lead time of each workpiece}: L = [l_1, l_2, \ldots l_n] \quad (5)$$

$$\text{Capability of each machine}: \Xi = \begin{bmatrix} \xi_{11} & \cdots & \xi_{1m} \\ \vdots & \ddots & \vdots \\ \xi_{n1} & \cdots & \xi_{nm} \end{bmatrix} \quad (6)$$

where $\xi_{ij} = 1$ indicates that workpiece i can be processed on machine j. Otherwise $\xi_{ij} = 0$.

Energy consumed by each machine to process one unit of each workpiece:

$$E = \begin{bmatrix} e_{11} & \cdots & e_{1m} \\ \vdots & \ddots & \vdots \\ e_{n1} & \cdots & e_{nm} \end{bmatrix} \quad (7)$$

$$\text{Scheduling solution}: X = \begin{bmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nm} \end{bmatrix} \quad (8)$$

where x_{ij} represents the amount of workpiece i on machine j.

From the above equations, the mathematical model of an energy saving scheduling problem is established as follows.

$$\text{Find the minimum } E_c, \text{ where the total energy consumption } E_c = \sum_{i=1}^{n} \Sigma_{j=1}^{m} x_{ij} e_{ij}, \quad (9)$$

such that

$$n_i = \Sigma_{j=1}^{m} x_{ij} \quad (10)$$

$$x_{ij} \% d_1 = 0, \text{ where \% indicates the remainder operater.} \quad (11)$$

$$l_i \geq \text{maximum}\{t^j_{1\_to\_i} | t^j_{1\_to\_i} = \Sigma^i_{k=1} t_k x_{kj} + t^j_a, \quad 1 \leq j \leq m\},$$

where $t^j_{1\_to\_i}$ is the time used by machine j to finish processing workpiece 1 to workpiece i. (12)

$$x_{ij} = 0, \quad \text{if } \xi_{ij} = 0 \quad (13)$$

4.2. Genetic algorithm-based method

In this genetic algorithm-based method, a scheduling solution $\begin{bmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nm} \end{bmatrix}$ is a chromosome, whereas a row vector $[x_{i1}, \ldots, x_{im}]$ is a gene of the chromosome.

As shown in Fig. 13, first of all, 50 parent chromosomes are randomly generated. Afterward, for the crossover operation, two parent chromosomes are randomly chosen via the roulette-wheel selection algorithm [45]. A crossover position is then randomly determined. Afterward, two child chromosomes are created by duplicating the two parent chromosomes, and exchanging the genes at the crossover position. In addition, n mutation indices, each of which is between 0 and 1, are randomly generated for n genes. Values of any genes with a mutation index less than 0.1 have to be replaced by new values randomly generated. Subsequently, the amount of child chromosomes is checked. The crossover and mutation operations will continue until 50 child chromosomes are created. Afterward, a total of 100 chromosomes, referred to as "one generation" in this paper, are sorted via their fitness function values in descending order. The fitness function is Eq. (9). The top 50 chromosomes are chosen to be the parent chromosomes of the next generation. The evolution procedure will start over until the next generation reaches the 3001st generation, whose topmost parent chromosome contains the optimal manufacturing process parameters.

In addition, as far as the majority of manufacturing companies are concerned, the main emphasis is to maximize the productivity, along with the energy saving as the secondary emphasis. Maximizing the productivity may indicate minimizing the makespan or work-in-process (WIP). In order to satisfy the two emphases, a multi-objective optimization problem needs to be solved. Assume that the energy saving job shop scheduling problem has two objectives, i.e., to minimize both the makespan and energy consumption. To solve this multi-objective optimization problem, the only thing that has to be changed in our framework is the fitness function of the genetic algorithm-based method. The fitness function now consists of two parts, one of which relates to the total makespan, and the other relates to the total energy consumption. As for the total makespan, it can be considered as the maximum time period, among all the m time periods, required by a specific machine to finish processing workpiece 1 to workpiece n. Therefore, similar to Eq. (12), the total makespan can be defined as follows.

$$t^{\max}_{1\_to\_n} = \text{maximum}\{t^j_{1\_to\_n} = \Sigma^n_{k=1} t_{kj} + t^j_a, \quad 1 \leq j \leq m\} \quad (14)$$

Moreover, in our genetic algorithm-based method, after the crossover and mutation operations, 100 solutions (parent and child chromosomes in total) will be produced before the fitness evaluation. Each solution corresponds to a total makespan $t^E_{1\_to\_n}$ and a total energy consumption E_c. Therefore, each solution's makespan and energy consumption can be normalized using the maximum and minimum makespans and energy consumptions, respectively, as follows.

$$\hat{t}^{\max_r}_{1\_to\_n} = (T_{\max} - t^{\max_r}_{1\_to\_n})/(T_{\max} - T_{\min}), \quad 1 \leq r \leq 100 \quad (15)$$

where

$$T_{\max} = \text{maximum}\{t^{\max_r}_{1\_to\_n} | 1 \leq r \leq 100\} \quad (16)$$

$$T_{\min} = \text{minimum}\{t^{\max_r}_{1\_to\_n} | 1 \leq r \leq 100\} \quad (17)$$

$$E^r_c = \frac{\varepsilon_{\max} - E^r_c}{\varepsilon_{\max} - \varepsilon_{\min}}, \quad 1 \leq r \leq 100 \quad (18)$$

Please cite this article in press as: Chou Y-C, et al. A bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling. J Manuf Syst (2013), http://dx.doi.org/10.1016/j.jmsy.2013.01.005

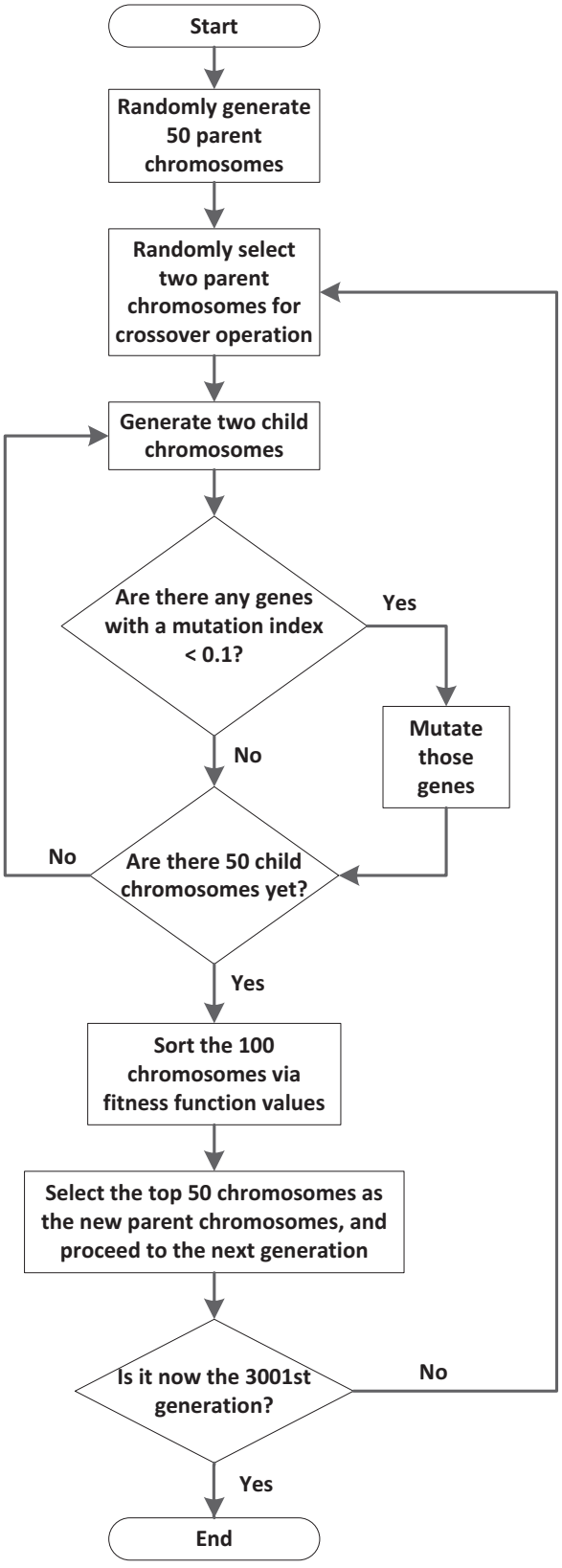

Fig. 13. A genetic algorithm-based method to find the optimal scheduling solution.

where

$$\varepsilon_{\max} = \text{maximum}\{E_c^r | 1 \le r \le 100\} \tag{19}$$

$$\varepsilon_{\min} = \text{minimum}\{E_c^r | 1 \le r \le 100\} \tag{20}$$

As shown in Eq. (15), a smaller makespan corresponds to a larger normalized value. The same conclusion can be drawn from Eq. (18) for the energy consumption as well. Eventually, in this multi-objective optimization problem, the function used to evaluate each solution's fitness can be defined as follows.

$$f_r = \alpha \times \hat{t}_{1\_to\_n}^{\max_r} + (1-\alpha) \times \widehat{E}_c^r, \quad 1 \le r \le 100, \quad 0 \le \alpha \le 1 \tag{21}$$

In Eq. (21), α is a pre-defined ratio used to reflect the relative importance of the two objectives, that is, minimizing the makespan and minimizing the energy consumption, in different application scenarios.

A job shop scheduling simulation that considers only the energy saving will be illustrated in the next section to validate the prototype system's self-configuration property.

4.3. Simulation and results

In this example, there are one scheduling center and two manufacturing cells. A scheduling center agency (SCY) is running at the scheduling center and a manufacturing cell agency (MCY) is running at each manufacturing cell. Meanwhile, a scheduling center agent (SCT) is running inside the SCY, whereas a manufacturing cell agent (MCT) is running inside each MCY. The example is simulated in a Linux environment. Here, a new order, containing six kinds of workpieces to be processed, is received by the SCT and needs to be assigned to a manufacturing cell. Each of the two manufacturing cells has five machines with different capabilities for processing those six kinds of workpieces.

Here, the self-configuration property of the system, or the decision-making ability of agents, to be demonstrated consists of two parts: (1) the order needs to be assigned to the most capable cell for manufacturing; (2) after the order is assigned to a manufacturing cell, the amount of each workpiece on each machine shall be arranged so that the least total energy will be consumed, which can be mathematically described as follows. With $E_{\text{consumption}} = \begin{bmatrix} e_{11} & \cdots & e_{15} \\ \vdots & \ddots & \vdots \\ e_{61} & \cdots & e_{65} \end{bmatrix}$, where e_{ij} represents the energy consumed by workpiece i on machine j, and all the necessary assumptions and constraints, find $X_{\text{schedule}} = \begin{bmatrix} X_{11} & \cdots & X_{15} \\ \vdots & \ddots & \vdots \\ X_{61} & \cdots & X_{65} \end{bmatrix}$, where X_{ij} represents the amount of workpiece i on machine j, such that $\text{minimum}(E_{\text{total\_consumption}} = \Sigma_{i=1}^{6} \Sigma_{j=1}^{5} X_{ij} e_{ij})$.

Fig. 14 shows the screenshot of runtime situations at the SCY. After the SCT starts at the SCY, it creates two BTs for the MCY_1 and MCY_2. Then the SCT waits for responses from the two BTs. As shown in the figure, the ACL message sent by a BT contains a value as its content. This value is the bid created by a BT to denote the capability of a manufacturing cell to process the order. As shown in the figure, the bid from the MCY_1 is larger than the one from the MCY_2, and therefore the SCT creates an OT for the MCY_1. The created OT will migrate to the MCY_1 to perform optimal scheduling.

Figs. 15 and 16 show the screenshots of runtime situations at the MCY_2 and MCY_1, respectively, for interactions between the BT and local MCT. As shown in both figures, the BT sends an ACL message to the MCT to request the capability matrix for all the machines in the cell. The capability matrix is formed based on the six kinds of workpieces. The exchanges of information regarding

```
File Edit View Terminal Help
ycchou@dragon:~/Papers/MESA10/demo$ ./scheduling_center_agency

Scheduling Center Agency (SCY) started

Scheduling Center Agent (SCT) started at SCY

SCT:
  Created Bidder Agents (BT) for MCY_1 and MCY_2

SCT:
  Received an ACL message:
    From:
      BT at MCY_2
    Content:
      19

SCT:
  Received an ACL message:
    From:
      BT at MCY_1
    Content:
      22

SCT:
  Created Order Agent (OT) for MCY_1
```

Fig. 14. Screenshot of runtime situations at the SCY.

the workpieces between the BT and MCT are not shown in this paper. In the capability matrix here, if element (2, 3) equals 1, it means that machine #3 is able to process workpiece #2. Each kind of workpiece has a different amount. Once the BT receives the capability matrix, it generates a bid that is an overall capability index for the local manufacturing cell. Here, for an illustrative purpose, a bid is simply generated through a LCS (Longest Common Subsequence) based algorithm. After a bid is generated, the BT informs the SCT of this capability index through an ACL message.

Fig. 17 shows the screenshot of runtime situations at the MCY_1 where the OT interacts with the MCT for necessary information to

```
File Edit View Terminal Help
ycchou@dragon:~/Papers/MESA10/demo$ ./manufacturing_cell_agency_2

Manufacturing Cell Agency 2 (MCY_2) started

Manufacturing Cell Agent (MCT) started

Bidder Agent (BT) arrived and started

BT:
  Sent an ACL message to MCT

MCT:
  Received an ACL message:
    From:
      BT
    Content:
      Capability matrix

MCT:
  Sent an ACL message to BT

BT:
  Received an ACL message:
    From:
      MCT
    Content:
      1 0 1 1 0
      0 1 1 0 0
      1 0 1 1 0
      1 1 1 1 0
      1 0 1 1 0
      1 1 1 1 0

BT:
  Generated a bid: 19
  Sent an ACL message to SCT
```

Fig. 15. Screenshot of runtime situations at the MCY_2 for interactions between the BT and local MCT.

```
File Edit View Terminal Help
ycchou@dragon:~/Papers/MESA10/demo$ ./manufacturing_cell_agency_1

Manufacturing Cell Agency 1 (MCY_1) started

Manufacturing Cell Agent (MCT) started

Bidder Agent (BT) arrived and started

MCT:
  Received an ACL message:
    From:
      BT
    Content:
      Capability matrix

MCT:
  Sent an ACL message to BT

BT:
  Sent an ACL message to MCT

BT:
  Received an ACL message:
    From:
      MCT
    Content:
      1 0 1 1 0
      0 1 1 0 1
      1 0 1 1 0
      1 1 1 1 1
      1 0 1 1 0
      1 1 1 1 1

BT:
  Generated a bid: 22
  Sent an ACL message to SCT
```

Fig. 16. Screenshot of runtime situations at the MCY_1 for interactions between the BT and local MCT.

perform the optimal schedule searching. As shown in the figure, the OT carries order data from the SCY. The order data are presented as the quantity vector, lead time vector, and process time vector. Here, for an illustrative purpose, these kinds of information are defined as

```
File Edit View Terminal Help
Order Agent (OT) arrived and started:
  Carrying order data:
    Quantity vector: 20 30 40 50 60 50
    Lead time vector: 2.0 7.0 3.0 10.0 4.5 5.0
    Process time vector: 120.0 110.0 80.0 120.0 45.0 60.0

OT:
  Sent an ACL message to MCT

MCT:
  Received an ACL message:
    From:
      OT
    Content:
      Begin time vector
      Power vector
      Power variation vector
      Capability matrix

MCT:
  Sent an ACL message to OT

OT:
  Received an ACL message:
    From:
      MCT
    Content:
      0.5 0 0 1 0

      7.5 12 7.5 15 12

      0.05 -0.02 0.08 -0.01 0.15

      1 0 1 1 0
      0 1 1 0 1
      1 0 1 1 0
      1 1 1 1 1
      1 0 1 1 0
      1 1 1 1 1
```

Fig. 17. Screenshot of runtime situations at the MCY_1 for interactions between the OT and local MCT for scheduling related information.

Please cite this article in press as: Chou Y-C, et al. A bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling. J Manuf Syst (2013), http://dx.doi.org/10.1016/j.jmsy.2013.01.005

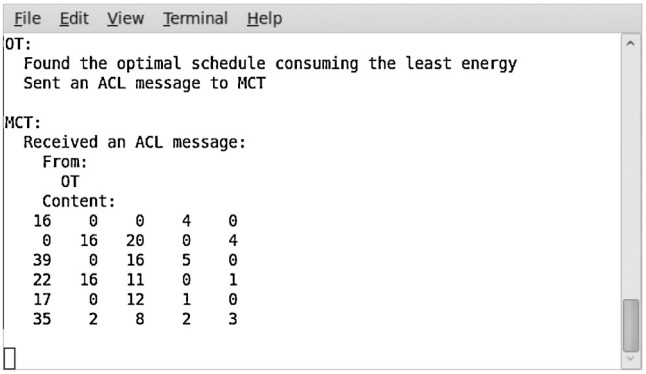

Fig. 18. Screenshot of runtime situations at the MCY_1 where the OT obtains the optimal schedule and informs the MCT of this schedule.

follows. An element of the quantity vector (in batches) represents the amount of one kind of workpiece. An element of the lead time (in days) vector represents the maximum allowable make-span for the entire amount of one kind of workpiece. An element of the process time vector (in minutes) represents the time needed to be spent on one batch of one kind of workpiece. As shown in the figure, the OT requests four kinds of information presented as the begin time vector, power vector, power variation vector, and capability matrix. An element of the begin time vector (in days) represents the time period for which a ma chine is still unavailable. An element of the power vector (in kW) represents the power that maintains a machine to stay in ready status. An element of the power variation vector represents the variation coefficient of that machine due to its operating conditions.

Fig. 18 shows the screenshot of runtime situations at the MCY_1 where the OT obtains the optimal job shop schedule, $\begin{bmatrix} 16 & 0 & 0 & 4 & 0 \\ 0 & 16 & 20 & 0 & 4 \\ 39 & 0 & 16 & 5 & 0 \\ 22 & 16 & 11 & 0 & 1 \\ 17 & 0 & 12 & 1 & 0 \\ 35 & 2 & 8 & 2 & 3 \end{bmatrix}$, that consumes the least total energy of MCY_1. The OT informs the MCT of this optimal schedule through an ACL message, as shown in the figure. Here, a genetic algorithm-based searching method is used to find the optimal job shop schedule. The evolutionary trend for the energy consumption at the MCY_1 is shown in Fig. 19. At the first generation, the calculated energy consumption is 3455.5 kW. Starting from the 2533rd generation, the calculated energy consumption remains as 3118.23 kW, meaning that an optimal solution has already been found, which leads to a 10% reduction in the energy consumption. Additionally, the figure also reveals a deterministic trend of reduction in the energy consumption.

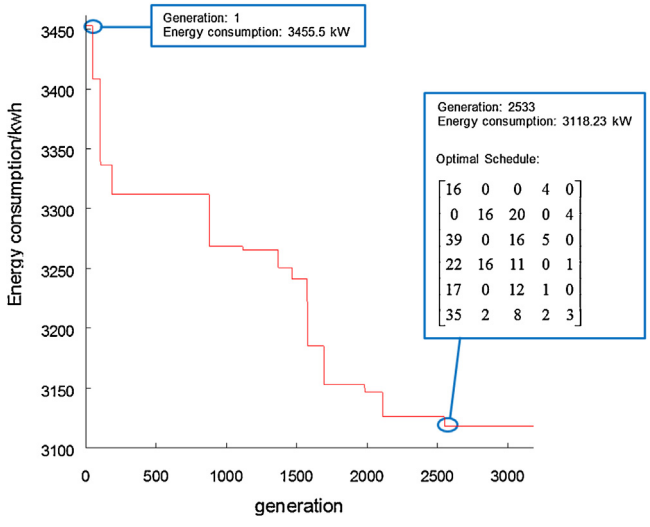

Fig. 19. Evolutionary trend of the energy consumption at the MCY_1.

5. Conclusion

This paper presents a bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling. An overall architecture of the system and brief implementation descriptions about the agency and agents in the system have been presented in this article. An example for energy saving job shop scheduling has been used to validate the system's self-configuration property. The self-configuration property demonstrated in this article is comprised of: (1) the order is automatically assigned to the most capable cell for manufacturing purposes; (2) after the order is assigned to a manufacturing cell, the amount of each workpiece on each machine is automatically arranged so that the total energy consumption of the manufacturing cell reaches the minimum.

In addition, the presented system has features that can create some advantages over other mobile agent-based manufacturing systems or mechanisms. The system matches the autonomic system reference architecture proposed by IBM and has the four IBM-defined fundamental self-managing properties. This system can thus manage itself with little human intervention. The system conforms to the IEEE FIPA standard. The community of FIPA users is growing in both the academia and industry. Therefore, this compliance ensures the interoperability between the presented system's agents and other agents from a variety of FIPA compliant agent platforms. The system supports the execution of C/C++ mobile agent codes. The system is thus applicable to a wide range of applications, due to C/C++'s comprehensive functionality, broad use, and international standard.

References

[1] Fuggetta A, Picco GP, Vigna G. Understanding code mobility. IEEE Transactions on Software Engineering 1998;24:342–61.
[2] Shen W, Norrie DH, Barthes JP. Multi-agent systems for concurrent intelligent design and manufacturing. London, UK: Taylor & Francis; 2001.
[3] Jennings NR. An agent-based approach for building complex software systems—why agent-oriented approaches are well suited for developing complex, distributed systems. Communications of the ACM 2001;44:35–41.
[4] Ouelhadj D, Petrovic S. A survey of dynamic scheduling in manufacturing systems. Journal of Scheduling 2009;12:417–31.
[5] Shen W, Wang L, Hao Q. Agent-based distributed manufacturing process planning and scheduling: a state-of-the-art survey. IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews 2006;36:563–77.
[6] Shen W. Distributed manufacturing scheduling using intelligent agents. IEEE Intelligent Systems 2002;17:88–94.
[7] Kirkpatrick S, Gelatt CD, Vecchi MP. Optimization by simulated annealing. Science 1983;220:671–80.
[8] Goldberg DE. Genetic algorithms in search, optimization, and machine learning. 1st ed. Boston, Massachusetts, USA: Addison-Wesley Professional; 1989.
[9] Shen W. Genetic algorithms in agent-based manufacturing scheduling systems. Integrated Computer-Aided Engineering 2002;9:207–17.
[10] Boccalatte A, Gozzi A, Paolucci M, Queirolo V, Tamoglia M. A multi-agent system for dynamic just-in-time manufacturing production scheduling, in. IEEE International Conference on Systems, Man, and Cybernetics 2004:5548–53.
[11] Li Y, Li S-J, Liu Y, Liu Z-G, Tang J. Dynamic scheduling method based on combination of contract net with mediator. In: Proceedings of 2005 international conference on machine learning and cybernetics. 2005. p. 339–344.
[12] Yu X, Ram B. Bio-inspired scheduling for dynamic job shops with flexible routing and sequence-dependent setups. International Journal of Production Research 2006;44(November):4793–813.
[13] Xiang W, Lee HP. Ant colony intelligence in multi-agent dynamic manufacturing scheduling. Engineering Applications of Artificial Intelligence 2008;21:73–85.
[14] Wei Y, Gu K, Liu H, Li D. Contract net based scheduling approach using interactive bidding for dynamic job shop scheduling. In: IEEE international conference on integration technology. 2007. p. 281–286.

[15] Madureira A, Gomes N, Santos J, Ramos C. Cooperation mechanism for team-work based multi-agent system in dynamic scheduling through meta-heuristics. In: IEEE international symposium on assembly and manufacturing. 2007. p. 233–238.
[16] Madureira A, Santos J, Gomes N, Ferreira I. Developing a multi-agent system for dynamic scheduling trough AOSE perspective. In: Elleithy K, editor. Advances and innovations in systems, computing sciences and software engineering. Netherlands: Springer; 2007. p. 35–40.
[17] Kang K, Zhang R, Yang Y. MAS equipped with ant colony applied into dynamic job shop scheduling. In: Huang D-S, Heutte L, Loog M, editors. Advanced intelligent computing theories and applications. With aspects of artificial intelligence, vol. 4682. Berlin/Heidelberg: Springer; 2007. p. 823–835.
[18] Rajabinasab A, Mansour S. Dynamic flexible job shop scheduling with alternative process plans: an agent-based approach. The International Journal of Advanced Manufacturing Technology 2011;54:1091–107.
[19] Asadzadeh L, Zamanifar K. An agent-based parallel approach for the job shop scheduling problem with genetic algorithms. Mathematical and Computer Modelling 2010;52:1957–65.
[20] Ennigrou M, Ghédira K. New local diversification techniques for flexible job shop scheduling problem with a multi-agent approach. Autonomous Agents and Multi-Agent Systems 2008;17:270–87.
[21] Aydin ME, Fogarty TC. A simulated annealing algorithm for multi-agent systems: a job-shop scheduling application. Journal of Intelligent Manufacturing 2004;15:805–14.
[22] Hao Q, Shen W, Wang L. Collaborative manufacturing resource scheduling using agent-based web services. International Journal of Manufacturing Technology and Management 2006;9:309–27.
[23] Papaioannou T, Edwards J. Manufacturing systems integration and agility: can mobile agents help? Integrated Computer-Aided Engineering 2001;8:68–80.
[24] Wong TN, Fang F. A multi-agent protocol for multilateral negotiations in supply chain management. International Journal of Production Research 2010;48:271–99.
[25] Stefano AD, Santoro C. A3M: an agent architecture for automated manufacturing. Software-Practice & Experience 2009;39:137–62.
[26] Huang C-Y, Cheng K, Holt A. An integrated manufacturing network management framework by using mobile agent. International Journal of Advanced Manufacturing Technology 2007;32:822–33.
[27] Nasshehi A, Allen RD, Newman ST. Application of mobile agents in interoperable STEP-NC compliant manufacturing. International Journal of Production Research 2006;44:4159–74.
[28] Wang X, Jiang A-G, Wang S. Mobile agent based wireless sensor network for intelligent maintenance. Lecture Notes in Computer Science 2005;3645:316–25.
[29] Hadeli K, Valckenaers P, Zamfirescu C, Brussel HV, Germain BS, Hoelvoet T, Steegmans E. Self-organizing in multi-agent coordination and control using stigmergy. Lecture Notes in Computer Science 2004;2977:105–23.
[30] Shin M-S, Jung M-Y. MANPro: mobile agent-based negotiation process for distributed intelligent manufacturing. International Journal of Production Research 2004;42:303–20.
[31] Zhou G-H, Jiang P-Y, Fukuda S. Using mobile agents to schedule a manufacturing chain on the internet. Concurrent Engineering: Research and Applications 2002;10:311–23.
[32] IBM. Autonomic computing white paper—an architectural blueprint for autonomic computing; June 2006.
[33] Kephart JO, Chess DM. The vision of autonomic computing. IEEE Computer 2003;36:41–50.
[34] FIPA. The Foundation for Intelligent Physical Agents. http://www.fipa.org/repository/standardspecs.html
[35] Chen B, Cheng HH, Palen J. Mobile-C: a mobile agent platform for mobile C/C++ code. Software – Practice & Experience 2006;36:1711–33.
[36] Chou Y-C, Ko D, Cheng HH. Mobile agent-based computational steering for distributed applications. Concurrency and Computation: Practice and Experience 2009;21:2377–99.
[37] Chou Y-C, Ko D, Cheng HH. An embeddable mobile agent platform supporting runtime code mobility, interaction and coordination of mobile agents and host systems. Information and Software Technology 2010;52:185–96.
[38] Chen B, Linz DD, Cheng HH. XML-based agent communication, migration and computation in mobile agent systems. Journal of Systems and Software 2008;81:1364–76.
[39] Cheng HH. Scientific computing in the Ch programming language. Scientific Programming 1993;2:49–75.
[40] Cheng HH. Ch: a C/C++ interpreter for script computing. C/C++ User's Journal 2006;24:6–12.
[41] Larson J, Cheng H. Object-oriented cam design through the internet. Journal of Intelligent Manufacturing 2000;11:515–34.
[42] China Energy Conservation Online. http://www.cecol.com.cn/a/list_1344.html
[43] Cao H, Liu F, He Y. Integrated task-assigning model of energy saving and noise reduction in the machining systems and its application. Journal of Mechanical Engineering 2006;42:97–102.
[44] Cao H, Tao X, Liu F. An energy saving scheduling model for machine tools and its application. Mechanical Science and Technology for Aerospace Engineering 2010;29:744–8.
[45] Fogel DB. Evolutionary computation: toward a new philosophy of machine intelligence. Hoboken, New Jersey, USA: Wiley-IEEE Press; 2006.

Please cite this article in press as: Chou Y-C, et al. A bio-inspired mobile agent-based integrated system for flexible autonomic job shop scheduling. J Manuf Syst (2013), http://dx.doi.org/10.1016/j.jmsy.2013.01.005