NRC Publications Archive Archives des publications du CNRC

Distributed scheduling for reactive maintenance of complex systems

Wang, C.; Ghenniwa, H.; Shen, W.

This publication could be one of several versions: author's original, accepted manuscript or the publisher's version. / La version de cette publication peut être l'une des suivantes : la version prépublication de l'auteur, la version acceptée du manuscrit ou la version de l'éditeur.

For the publisher's version, please access the DOI link below. / Pour consulter la version de l'éditeur, utilisez le lien DOI ci-dessous.

Publisher's version / Version de l'éditeur:

<https://doi.org/10.1109/TSMCC.2007.913916>

IEEE Transactions on Systems, Man, and Cybernetics, Part C, 38, 2, pp. 269-273, 2008-03-01

NRC Publications Record / Notice d'Archives des publications de CNRC:

<https://nrc-publications.canada.ca/eng/view/object/?id=840d8f42-8bb3-4561-9290-eb58b4e88c9e>

<https://publications-cnrc.canada.ca/fra/voir/objet/?id=840d8f42-8bb3-4561-9290-eb58b4e88c9e>

Access and use of this website and the material on it are subject to the Terms and Conditions set forth at

<https://nrc-publications.canada.ca/eng/copyright>

READ THESE TERMS AND CONDITIONS CAREFULLY BEFORE USING THIS WEBSITE.

L'accès à ce site Web et l'utilisation de son contenu sont assujettis aux conditions présentées dans le site

<https://publications-cnrc.canada.ca/fra/droits>

LISEZ CES CONDITIONS ATTENTIVEMENT AVANT D'UTILISER CE SITE WEB.

Questions? Contact the NRC Publications Archive team at

PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca. If you wish to email the authors directly, please see the first page of the publication for their contact information.

Vous avez des questions? Nous pouvons vous aider. Pour communiquer directement avec un auteur, consultez la première page de la revue dans laquelle son article a été publié afin de trouver ses coordonnées. Si vous n'arrivez pas à les repérer, communiquez avec nous à PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca.

<http://irc.nrc-cnrc.gc.ca>

Distributed scheduling for reactive maintenance of complex systems

NRCC-50278

Wang, C.; Ghenniwa, H.; Shen, W.

March 2008

A version of this document is published in / Une version de ce document se trouve dans:
IEEE Transactions on Systems, Man, and Cybernetics, Part C, v. 38, no. 2, 2008, pp 269-273

The material in this document is covered by the provisions of the Copyright Act, by Canadian laws, policies, regulations and international agreements. Such provisions serve to identify the information source and, in specific instances, to prohibit reproduction of materials without written permission. For more information visit <http://laws.justice.gc.ca/en/showtdm/cs/C-42>

Les renseignements dans ce document sont protégés par la Loi sur le droit d'auteur, par les lois, les politiques et les règlements du Canada et des accords internationaux. Ces dispositions permettent d'identifier la source de l'information et, dans certains cas, d'interdire la copie de documents sans permission écrite. Pour obtenir de plus amples renseignements : <http://lois.justice.gc.ca/fr/showtdm/cs/C-42>

National Research
Council Canada

Conseil national
de recherches Canada

Canada

Distributed Scheduling for Reactive Maintenance of Complex Systems

Chun Wang, *Student Member, IEEE*, Hamada Ghenniwa, and Weiming Shen, *Senior Member, IEEE*

Abstract—This paper presents a distributed scheduling algorithm for reactive maintenance of complex systems. The proposed algorithm uses an iterative bidding procedure to assign operations of maintenance jobs to engineers with partially overlapped skill sets. In each round, an unassigned operation is selected based on the overlapping degree of engineers' skill sets on the operation and its average processing time among capable engineers. Engineers' availability and cost information are used to determine the winner of an assignment. The effectiveness of this approach is demonstrated through a computational case study. Prototype implementation and applications to real world domains are discussed.

Index Terms—Distributed scheduling, reactive maintenance, multiagent system.

I. INTRODUCTION

In many organizations, maintenance can be generally seen as the function of sustaining the integrity of equipment by repairing, modifying or replacing them as necessary. Among various maintenance environments, we consider a specific class in which the systems to be maintained are complex systems, such as airplanes, power plants, healthcare facilities, or automated manufacturing systems. Due to the high complexity of these systems, effective maintenance management is essential in maintaining system availability and reducing operational costs. From the perspective of maintenance management, complex systems maintenance has unique characteristics that differentiate it from other environments:

1) *Diversity of equipment and high level of technologies*: Complex systems usually consist of diverse subsystems and components. A typical flexible manufacturing system (FMS) contains several types of computer numerical controlled (CNC) machines, sophisticated material handling and control systems. In many manufacturing plants, the equipment was specially made and does not exist anywhere else. In addition, today's complex systems are equipped with various advanced technologies. A typical process plant such as an oil refinery is fully automated. Almost all the equipment has microprocessor controls, self diagnostics, and interfaces. In such a diverse and high technology environment, it may take years for an engineer to be effectively able to perform just a portion of all maintenance operations required by a system. Maintenance engineers' skill sets become more specialized.

2) *High cost of downtime*: Complex systems are usually

capital-intensive and/or mission critical. A system shutdown generally inflicts a high cost. As reported in [3], the downtime cost for a cigarette manufacturer in the United States is over \$1,000 a minute per assembly line; a power utility may run up to \$5,000 or more a minute for outages; an automotive assembly line can cost \$4,000 per minute for downtime. It is clear that to be effective these characteristics need to be considered in complex systems maintenance management.

One of the important functionalities of maintenance management is scheduling, which allocates the maintenance jobs to available resources, such as engineers, overtime. A typical goal of scheduling in complex systems maintenance is to lower the cost by reducing the system downtime and controlling the service cost needed to perform the maintenance jobs. However, complex systems maintenance presents particular challenges for achieving this goal:

1) *Computational complexity*: Due to the diversity of equipment and high level of technologies, engineers are likely capable of performing specialized maintenance operations. Therefore, maintenance jobs consisting of different operations may require the collaboration of a team of engineers. Each engineer is responsible for one or several operations within his/her skill set, and in many cases the skill sets of engineers are partially overlapped. In order to reduce downtime, maintenance jobs on several components of the system are often executed concurrently. Thus the scheduling problem involves the allocation of a set of multi-operation maintenance jobs to engineers with partially overlapped capabilities, in a way that the overall system downtime and service cost are minimized. This problem is known as partially overlapped system scheduling problem [4], which is in general NP-hard [14].

2) *Limited computation time*: For proactive (planned) maintenance, a schedule can be computed upfront and the computation time does not have impact on the system downtime. However, emergencies happen. For reactive (unplanned) maintenance, the time of computing a schedule is counted in the system downtime. Due to the high downtime cost, we want to compute a schedule as soon as possible. The time that is allowed to compute the schedule is limited.

3) *Distributed environment*: Maintenance outsourcing is a fast growing business today. More and more maintenance jobs are being carried out by private engineers or engineers from other service companies. The environments are becoming distributed. As a result of the distributed environments, detailed information of engineers, such as their current

workloads, capabilities and availabilities during a certain period may not be available to the scheduler. Computing schedules without a global view of the environment becomes a challenge.

This correspondence proposes an agent-based distributed scheduling approach for maintenance job scheduling in a reactive maintenance environment. Particularly, we demonstrate how this approach addresses the challenges mentioned above. The rest of the correspondence is organized as follows. Section II presents a complex system maintenance scheduling problem formulation. Section III proposes an agent-based distributed maintenance scheduling approach. Section IV discusses the application of the proposed approach in real world domains. Section V reviews related work. Section VI concludes the paper.

II. PROBLEM FORMULATION

To clearly demonstrate the combinatorial optimization nature of the problem, we assume a centralized environment in this section, i.e., all the scheduling related information is available to the scheduler. With this assumption, we can conveniently model the problem as a mixed integer program. The distributed environment will be considered when we develop the agent-based scheduling algorithm in Section III.

A. Problem Description

Consider a complex system with a set of maintenance jobs to be executed. Each job j ($j=1, \dots, n$) requires the processing of a sequence of operations $o_{j,k}$ ($k=1, \dots, n_j$).

Let O denote the set of all operations, $O = \bigcup_{j=1}^n \bigcup_{k=1}^{n_j} o_{j,k}$. Maintenance jobs are carried out by a group of engineers. Each engineer i ($i=1, \dots, m$) can perform a set of operations, denoted by M_i , and their capabilities are partially overlapped, that is, for some $i, \hat{i}$ ($1 \leq i \leq m, 1 \leq \hat{i} \leq m, i \neq \hat{i}$), $M_i \cap M_{\hat{i}} \neq \emptyset$. For an operation $o_{j,k}$, if an engineer i is capable of processing it, a processing time, $p_{i,j,k}$ and a labor cost, $q_{i,j,k}$, are given. $p_{i,j,k} = +\infty$, $q_{i,j,k} = +\infty$ if $o_{j,k}$ cannot be performed by engineer i . A job j can only begin after its release time r_j . There are precedence constraints among operations of a job which ensure the processing order. No precedence constraints among jobs. There is no more than one operation performed by the same engineer at the same time. The scheduling problem involves the allocation of engineers to maintenance jobs such that, certain objective functions (will be defined later in this section) are minimized and all constraints are satisfied.

B. Definition of Variables

Variables used in the formulation are defined as follows.

$S_{j,k}$ starting time of $o_{j,k}$

$$X_{i,j,k} = \begin{cases} 1 & \text{if engineer } i \text{ is chosen to perform } o_{j,k} \\ 0 & \text{otherwise} \end{cases}$$

$$Y_{j,k,\hat{j},\hat{k}} = \begin{cases} 1 & \text{if } o_{j,k} \text{ is performed before} \\ & o_{\hat{j},\hat{k}} \text{ by the same engineer} \\ 0 & \text{otherwise, } j \neq \hat{j} \end{cases}$$

C. Constraints

We have modeled 8 sets of constraints for the problem:

$$S_{j,1} \geq r_j \quad j=1, \dots, n \quad (1)$$

$$S_{j,k-1} + \sum_{i=1}^m p_{i,j,k-1} X_{i,j,k-1} - S_{j,k} \leq 0 \quad j=1, \dots, n, \quad 1 < k \leq n_j \quad (2)$$

$$S_{j,k} + \sum_{i=1}^m p_{i,j,k} X_{i,j,k} - S_{\hat{j},\hat{k}} + HX_{i,j,k} + HX_{i,\hat{j},\hat{k}} + HY_{j,k,\hat{j},\hat{k}} \leq 3H, \quad (3)$$

$$j=1, \dots, n, \quad \hat{j}=1, \dots, n, \quad j \neq \hat{j}, \quad 1 \leq k \leq n_j, \quad 1 \leq \hat{k} \leq n_{\hat{j}} \quad (4)$$

$$Y_{j,k,\hat{j},\hat{k}} + Y_{\hat{j},\hat{k},j,k} = 1, \quad (5)$$

$$j=1, \dots, n, \quad \hat{j}=1, \dots, n, \quad j \neq \hat{j}, \quad 1 \leq k \leq n_j, \quad 1 \leq \hat{k} \leq n_{\hat{j}} \quad (6)$$

$$\sum_{i=1}^m X_{i,j,k} = 1, \quad j=1, \dots, n, \quad 1 \leq k \leq n_j \quad (7)$$

$$X_{i,j,k} \in \{0, 1\}, \quad j=1, \dots, n, \quad 1 \leq k \leq n_j \quad (8)$$

$Y_{j,k,\hat{j},\hat{k}} \in \{0, 1\}, \quad j=1, \dots, n, \quad \hat{j}=1, \dots, n, \quad j \neq \hat{j}, \quad 1 \leq k \leq n_j, \quad 1 \leq \hat{k} \leq n_{\hat{j}} \quad (9)$

$S_{j,k} \geq 0, \quad j=1, \dots, n, \quad 1 \leq k \leq n_j \quad (8)$

The set of constraints (1) ensures that a maintenance job does not start before its release time. The set of constraints (2) ensures that an operation does not start before the previous operation of the same job is completed. The set of constraints (3) and (4) ensure that at most one job can be processed by a particular engineer at a time. In (3) H is a large finite positive number. Constraints (5) say one operation can and only can be processed by one engineer. Constraints (6), (7), and (8) are non-negative and integer constraints.

D. Objective Functions

The objective of the maintenance job scheduling is to minimize the total maintenance cost which we model as the sum of the *system downtime* cost and the labor cost of engineers, formulated as follows:

$$\min \left(\bar{\lambda} \max_{j=1, \dots, n} \left(S_{j,n_j} + \sum_{i=1}^m p_{i,j,n_j} X_{i,j,n_j} \right) + \sum_{j=1}^n \sum_{i=1}^m \sum_{k=1}^{n_j} q_{i,j,k} X_{i,j,k} \right) \quad (9)$$

where $\bar{\lambda}$ denotes the cost of each unit of the *system downtime*.

III. AGENT-BASED DISTRIBUTED SCHEDULING

This section presents an agent-based distributed scheduling algorithm for reactive maintenance. The algorithm is designed in the context of a Multi-agent Collaborative Maintenance System (MCMS) architecture which was presented in [15]. In this architecture, five types of agents (Help desk agent, Engineer agent, Broker agent, Diagnostic agent, and Directory Facilitator) work collaboratively to achieve the overall maintenance functions of the system. The distributed scheduling algorithm proposed in this correspondence involves three types of agents: Engineer agent, Broker agent, and Director Facilitator. The Broker agent represents the maintenance manager and an Engineer agent functions as the personal assistant of an engineer. The Director Facilitator provides registration and lookup services for other agents. The design requirement for the agent-based scheduling algorithm is to quickly respond to failures of complex systems with low cost maintenance schedules.

A. Negotiation protocol

The distributed scheduling can be seen as a coordination process between the Broker agent and the Engineer agents, during which a negotiation protocol is used to allocate maintenance jobs to Engineers. The allocation is an iterative process. Each round of it is implemented using a contract-net [11] based protocol which contains the following three steps:

1) The Broker agent first selects an operation from the maintenance jobs to be scheduled. Then it solicits bids for completion of the operation from eligible Engineer agents (the Broker agent can find out which Engineer agent is eligible for the selected operation by looking up the Directory Facilitator).

2) Each eligible Engineer agent who is willing to participate sends a bid back to the Broker agent. Each bid specifies an offer of promised completion time and the cost at which the processing of the operation with the offered completion time is delivered.

3) Upon receiving bids, the Broker agent selects the best one from them according to its preferences and local constraints and assigns the operation to the winning Engineer agent. The procedure continues until all operations of maintenance jobs have been assigned to Engineer agents.

B. Broker Agent's Local Decision Making

At local level, the Broker agent has two decision making problems: (1) selecting an operation to be assigned out; (2) determining the winning Engineer agent to which the operation is assigned.

For operation selection, we propose a heuristic algorithm. The algorithm is a composite of several dispatching rules. The basic idea is to effectively utilize the flexibility provided by the partially overlapping characteristic of the capabilities of engineers to balance their work loads. Based on this heuristic, we propose a composite dispatching rule, called Flexible Operation Last (FOL). FOL is a composite of two elementary dispatching rules, Longest Processing Time first (LPT) and Least Flexible Job first (LFJ). As a composite dispatching

rule, FOL is modeled as a ranking expression that combines LPT and LFJ. This combination can be implemented as the following function: $f_{j,k} = \exp(u_{j,k}/Q)/l_{j,k}$ where $f_{j,k}$ is the flexibility ranking index of $o_{j,k}$, defined as the flexibility of operation k of job j ; $u_{j,k}$ is the percentage of engineers who are capable of performing $o_{j,k}$, we use $u_{j,k}$ to indicate the *overlapping degree* of engineers' capabilities on $o_{j,k}$; $l_{j,k}$ is the average processing time of the operation among eligible engineers. Q is the scaling parameter that can be determined empirically. If Q is very large, the FOL rule reduces to the LPT rule. If Q is very small, the rule reduces to the LFJ rule. During each round, the FOL selects an operation from the Eligible Operation Set¹ (EOS). Operations with higher flexibility ranking indices (with short average processing time and more eligible engineers) are placed towards the end of the schedule, where they can be used to balance work loads more effectively.

The winning Engineer agent is determined based on the bids submitted by the participated Engineer agents according to the Broker agent's utility function. The Broker agent derives utility from a bid $(C_{i,j,k}, q_{i,j,k}) \in \mathcal{R}_+^2$, which represents the assignment of operation k of job j to Engineer agent i with completion time $C_{i,j,k}$ and cost $q_{i,j,k}$. The utility function of the Broker agent is quasilinear and takes the form

$$U(C_{i,j,k}, q_{i,j,k}) = V(C_{i,j,k}) - q_{i,j,k}$$

where $V(C_{i,j,k})$ is a function defining the value that the Broker agent has on the bid with completion time $C_{i,j,k}$. $V(C_{i,j,k})$ may be constructed in different ways. For demonstration we present a construction of $V(C_{i,j,k})$ as follows.

$$V(C_{i,j,k}) = \begin{cases} \bar{\lambda}(T - C_{i,j,k}) & \text{if } C_{i,j,k} \geq C_{i,j,k-1} + p_{i,j,k}, \quad 1 < i, \\ \hat{i} < m, 1 < j < n, 1 \leq k \leq n_j \\ 0, & \text{otherwise} \end{cases}$$

where T is a upper bound of the makespans of valid schedules. T can be obtained by timetabling the operations of jobs as early as possible without slack and each operation is assigned to the engineer with longest processing time for this operation. This produces (with effort linear in the number of

operations) a valid schedule of length $T = \sum_{j=1}^J \sum_{k=1}^{n_j} p_{i^*,j,k}$,

where $1 < i^* < m$ and $p_{i^*,j,k} \geq p_{i,j,k}$, for all $1 < i < m$.

Clearly that, based on this construction of V , bids with earlier completion times and lower costs generate higher utility to the Broker agent. On the other hand, bids that violate

¹ Eligible Operation Set contains operations for which the job release time and operation preceding constraints are satisfied.

precedence constraints of jobs give no value to the Broker agent. Upon receiving bids, the Broker agent selects the winning Engineer agent based on a scoring function, denoted as $S: (C_{i,j,k}, q_{i,j,k}) \rightarrow \mathbb{R}^+$; more specifically, the operation is assigned to the Engineer agent whose bid achieves the highest value of $S(\cdot)$. The reason for $S(\cdot)$ is to prevent the utility function of the Broker agent (considered as the Broker agent's private information) from being exposed to the Engineer agents. $S(\cdot)$ is publicly known to the Engineer agents at the start of bidding and the true preference of the Broker agent is reflected in $S(\cdot)$, i.e., $S(C_{i,j,k}, q_{i,j,k}) = f(U(C_{i,j,k}, q_{i,j,k}))$, where f is a non-decreasing function, that is, for any i and $\hat{i}$, if $U(C_{i,j,k}, q_{i,j,k}) \geq U(C_{\hat{i},j,k}, q_{\hat{i},j,k})$, then $S(C_{i,j,k}, q_{i,j,k}) \geq S(C_{\hat{i},j,k}, q_{\hat{i},j,k})$. The procedure of assigning operations of maintenance jobs can be summarized in Fig. 1.

Fig. 1. The procedure of assigning operations

C. Engineer Agents' Local Decision Making

An Engineer agent calculates its bid on an operation based on three factors: (1) the availability of the engineer which is determined by the current workload; (2) the time needed to complete the operation; (3) the labor cost for completing the operation. We model the Engineer agents' behaviour as "price-taking". They always propose bids with highest scoring function value according to their situations in each round of the assignment.

D. A Computational Study

The proposed negotiation protocol and agent's local decision making schemes have been implemented in MCMS.

We have named the distributed scheduling algorithm Distributed Heuristic Scheduling (DHS). Six sets of FMS maintenance scheduling problem instances of different size and structure were randomly generated and solved using DHS. When generating a specific problem set, we first fix the numbers of maintenance jobs, operations in each job, and engineers, then, randomly assign engineers' capabilities (capable operations and corresponding processing times). We assume engineers' labor cost rates on a specific operation are identical². Table I summarizes the configuration of the test problem sets.

Table I. Configuration of the test problem sets

Problem set name	Number of jobs	Number of operations in each job	Number of engineers
1op-9job-5m	9	1	5
2op-6job-5m	6	2	5
3op-6job-5m	6	3	5
5op-4job-5m	4	5	5
7op-4job-5m	4	7	5
7op-15job-10m	15	7	10

A total of over 200 distributed maintenance scheduling problem instances were generated and solved by DHS. We solved each instance three times using 10^0 , 10^2 and 10^4 as the values for Q respectively and selected the best objective function value achieved for the performance evaluation. To provide a benchmark for evaluating the performance of DHS, we also generated the centralized relaxations of the distributed scheduling problems by using the same data sets but relaxing the constraint of the distribution of information. That is, for the centralized relaxations, the Broker agent has a global view of all information, thus it can apply a centralized optimal algorithm to the relaxations directly (The relaxations use the problem formulation developed in Section II). Because the optimal schedules of the centralized relaxations are the best solutions that a distributed scheduling algorithm can possibly generate, we use them as the benchmark for evaluating our DHS. We have used ILOG CPLEX 10.0 to solve the centralized relaxations. Also, $\bar{\lambda}$ is normalized to 1. Fig. 2 shows the performance of DHS over 5 different problem sets against the optimal solutions. It shows that DHS performs well with 1-op set (on average, DHS generates 2.5% more cost than the optimal algorithm), and the Cost(DHS) gradually increases when the number of operations in jobs increases.

For small problem instances CPLEX can usually find optimal solutions within 10 minutes. However, for larger size instances (such as those in 7op-15job-10m) CPLEX is not likely to find an optimal (even a feasible) solution within reasonable time. We have tried 14 instances in 7op-15job-10m using CPLEX. It did not find optimal solution for any of them

² Since the purpose of this computational study is to validate the effectiveness of the operation flexibility heuristic, we deliberately exclude the cost factor from the comparison.

within 10 hours. Fig. 3 shows the comparison of DHS and CPLEX over 7op-15job-10m alone time line. DHS solves any one of the instances in 7op-15job-10m within 5 seconds with an average cost of 70.615. CPLEX iteratively improve the feasible schedule along the time line. The average cost of schedules that CPLEX found within 7000 seconds is 74.222. Within certain time limits (7000 seconds in this experiment) DHS performs better than CPLEX with higher solution quality and a small fraction of computing time.

Fig. 2. Performance of DHS on different problem sets

Fig. 3. Comparison of DHS and CPLEX over 7op-15job-10m along time line implementation

IV. PROTOTYPE AND APPLICATION

The proposed distributed scheduling system is a fairly general framework which finds applications in many real world domains. In the Distributed Intelligent Systems Lab at The University of Western Ontario, we have applied the approach to the reactive maintenance scheduling for a team of mobile robots in the MCMS environment shown in Fig. 4. As mentioned in Section 3, the system consists of several agents (Help Desk Agent, Diagnostic agent, mobile robots (Nagami, maintained in the system). For example, sensors, 16 infrared sensors, Pentium-II computer, wireless LAN and connected to the Internet.

Fig. 4 Multiagent Collaborative Maintenance System implementation

system failures. Once the system failures are detected by the robots, they report them to Diagnostic agent using predefined problem status code. Based on the reports, Diagnostic agent works out a reactive maintenance plan and passes the plan to Broker agent for scheduling. Broker agent schedules the maintenance jobs through the iterative negotiation procedure with engineer agents which are located in iPAQ handheld computers each for an engineer. Engineer agents can obtain remote technical supports from the In House Engineers (who are normally system experts) through the Help Desk agent. Several reactive maintenance scheduling scenarios have been tested using this prototype. The multiagent system has been implemented on the Java Agent Development Framework (JADE, <http://jade.tilab.com>). FIPA ACL (<http://www.fipa.org>) is used for inter-agent communication. Implementation details can be found in [15].

The approach can also be applied to other maintenance domains. Take aircraft emergency maintenance as an example. Modern air planes are complex systems. The maintenance of these planes usually requires a team of engineers with different expertise. The structure of the engineers' capabilities is a partially overlapped one. In the case of emergency maintenance, the maintenance manager can use the proposed distributed maintenance scheduling system to sequence maintenance operations and schedule them to the engineers from various departments or other companies without knowing their detailed capability and availability information. In fact, some aviation maintenance providers, such as United Services (the maintenance division of United Airlines) provide global emergency service for airlines using a Global Emergency Maintenance (GEM) team available 24/7/365. In fact, due to the diversity of airplane models and their system complexity, every maintenance service provider has its capability restrictions and no team is really available 24/7/365 if it is not dedicated to only one plane. Very often a maintenance manager needs to coordinate the work of several service providers and engineers to reduce the down time and to minimize the costs. In these cases, the proposed system can be a helpful tool in terms of quickly providing a good quality maintenance schedule in a distributed environment.

As a supporting tool, various types of Computerized Maintenance Management Systems (CMMS) have been integrated into enterprises' maintenance management in recent years. CMMS allows easier access to masses of data and to levels of analysis not readily available manually. As the

amount of information exchanged and agents' local decision-making schemes. Contract-net is a popular protocol that has been embedded in many distributed scheduling algorithms. Baker [1] proposed a Market-Driven Contract Net for heterarchical agent-based scheduling in which a sequential bid propagation scheme is used to subcontract subassemblies of a product to resource agents in the system. Lin and Solberg [5] introduced a market-based mechanism into the contract-net based negotiation. A price system is used to facilitate the distributed search. In [9], Saad et al. evaluated the performance of two contract-net based scheduling mechanisms, namely Production Reservation where all the operations of a job are scheduled completely at the time when it arrives to the system and Single Step Production Reservation where operations are scheduled one at a time. Other typical distributed scheduling systems include AARIA [7], the N^* algorithm [2], the leveled commitment contracting protocol by Sandholm and Lesser [10], and the texture-based distributed constrained heuristic search by Sycara et al. [12]. The above mentioned algorithms assume a heterarchical system structure in which agents are used to represent each resource and job. We adopt a different structure, in which the broker agent represents all the maintenance jobs to be scheduled and the search is mediated by the broker agent. This structure is similar to that developed in [6].

Walker et al. [13] applied a holonic, multiagent systems approach to manufacturing job shop scheduling. It employed a sophisticated evolutionary algorithm for turning the schedule creation rules rather than the schedule itself. The approach is evaluated based on a benchmark environment containing four types of resources, each having non-overlapping capabilities. An important characteristic of the maintenance scheduling problem investigated in this correspondence is that the capabilities of engineers are partially overlapped. The uniqueness of the proposed algorithm is two fold: (1) it explicitly models the flexibility of scheduling operations to engineers in terms of the overlapping degree of the engineers' capabilities on an operation and its average processing time among eligible engineers; (2) and the flexibility information is used as a heuristic to direct the search.

In spite of the large body of literature in distributed scheduling, few of them is dedicated to the reactive maintenance environment. Quere et al. [8] proposed a distributed scheduling algorithm for complex system maintenance in a cooperative environment composed of several decision centers, taking into account communication times. Their emphasis is rather different from ours, since they focus on the dynamic aspect of scheduling repair and the impact of communication times on the responsiveness of the system.

VI. CONCLUSION

It is common in complex system maintenance that a maintenance job contains multiple operations and the capabilities of engineers are partially overlapped. The

distributed scheduling algorithm presented here has been designed to offer fast responses to reactive complex system maintenance scheduling in distributed environments. It can also be used in proactive maintenance scheduling to calculate a good quality first incumbent solution that other search techniques such as branch and bound, simulated annealing, and Tabu algorithm can start with.

REFERENCES

- [1] A.D. Baker, *Manufacturing Control with a Market-Driven Contract Net* PhD Thesis, Rensselaer Polytechnic Institute, 1991.
- [2] H.W. Chun, and R.Y.M. Wong, "N* - An Agent-based Negotiation Algorithm for Dynamic Scheduling and Rescheduling," In *Advanced Engineering Informatics*, Volume 17, no 1, pp.1-22, 2003.
- [3] L. Joel, *The hand book of maintenance management*, New York, N.Y. Industrial Press, 1997.
- [4] M. Kamel, and H. Ghenniwa, "Partially-Overlapped Systems: The Scheduling Problem," in *Design and Implementation of Intelligent Manufacturing Systems*, Parsaei, H. and Jamshidi, M. (Eds.), Prentice-Hall, pp. 241-274. 1995
- [5] G.Y. Lin, and J. J. Solberg, "Integrated Shop Floor Control Using Autonomous Agents," *IIE Transactions*, vol. 24, no. 3, pp. 57-71, 1992.
- [6] F. Maturana and D. H. Norrie, "Multiagent Mediator Architecture for Distributed Manufacturing", *Journal of Intelligent Manufacturing*, vol7, no. 4, pp. 257-27, 1996.
- [7] H.V.D. Parunak, A.D. Baker, and S.J. Clark, "The AARIA Agent Architecture: An Example of Requirements-Driven Agent-Based System Design," *Proc. 1st Int'l Conf. Autonomous Agents*, ACM Press, New York, 1997, pp. 482-483.
- [8] Y.L. Quere, M. Sevaux, C. Tahon, D. Trentesaux, "Reactive scheduling of complex system maintenance in a cooperative environment with communication times," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, Volume 33, no. 2, pp. 225-234, 2003.
- [9] A. Saad, K. Kawamura and G. Biswas, "Performance Evaluation of Contract Net-Based Heterarchical Scheduling for Flexible Manufacturing Systems," *Intelligent Autonomous and Soft Computing*, Vol. 3, no. 3, pp. 229-248, 1997.
- [10] T. W. Sandholm and V. R. Lesser, "Advantages of a leveled commitment contracting protocol," in *Proc. 13th National Conference on Artificial Intelligence (AAAI-96)*, pp. 126-133, July 1996.
- [11] R.G. Smith, "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver," *IEEE Trans. Computers*, vol. C-29, no. 12, pp. 1104-1113, 1980.
- [12] K. Sycara, S. Roth, N. Sadeh and M. Fox, "Distributed constrained heuristic search," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 21, no. 6, pp. 1446-1461.
- [13] S. S. Walker, R. W. Brennan, D. H. Norrie, "Holonic Job Shop Scheduling Using a Multiagent System," *IEEE Intelligent Systems* Vol.20, No. 1, pp. 50-57, 2005.
- [14] C. Wang, *Scheduling Multi-Operation Tasks in Partially Overlapping Systems*, Master's Thesis, The University of Western Ontario, 2004.
- [15] C. Wang, H. Ghenniwa, W. Shen, Y. Zhang, "Agent-Based Distributed Collaborative Monitoring and Maintenance in Manufacturing," in Camarinha-Matos, L.M. (Ed.), *Emerging Solutions for Future Management Systems*, Springer, pp. 129-138, 2004.