Appl Intell (2017) 47:1068–1086
DOI 10.1007/s10489-017-0935-y

A hybrid and scalable multi-agent approach for patient scheduling based on Petri net models

Fu-Shiung Hsieh[1]

Published online: 12 May 2017

Abstract Scheduling patients in a hospital is a challenging issue due to distributed organizational structure, dynamic medical workflows, variability of resources and the computational complexity involved. It calls for a sustainable architecture and a flexible scheduling scheme that can dynamically allocate available resources to promptly react to patients in a hospital and deliver healthcare services timely. The objectives of this paper are to propose a viable and systematic approach to develop a scalable and sustainable scheduling system based on multi-agent system (MAS) to shorten patient stay in a hospital and plan schedules based on the medical workflows and available resources. To develop a patient scheduling system, we combine MAS architecture, contract net protocol (CNP), workflow specification models based on Petri nets and the cooperative distributed problem solving concept. To achieve interoperability and sustainability, Petri Net Markup Language (PNML) and XML are used to specify precedence constraints of operations in medical workflows and capabilities of resource agents, respectively. Agent communication language (ACL) and CNP are used to achieve communication and negotiation/mutual selection of agents. A collaborative algorithm is invoked by individual agents to optimize the schedules locally based on a problem formulation automatically obtained by Petri net models. We have developed a scheduling system based on a FIPA compliant MAS platform to solve the dynamic patient scheduling problem. To illustrate the benefit of our approach, we compare the performance of our method with a heuristic rule commonly used in practice. In addition, we also analyze and verify scalability of our approach by experiments.

Keywords Multi-agent system · Scheduling · Workflow · Resource allocation · Scalability

1 Introduction

The distributed organizational structure, costly infrastructure, diversity of medical processes, decreasing budget in public health insurances and the need for prompt response to patients pose challenges in management of hospitals. These challenges call for the development of an effective methodology to reduce time and cost and provide quality services based on a flexible distributed organizational architecture. The issues in hospitals range from allocation of resources, scheduling of patients to delivery of healthcare services. In existing literature, there are several studies on planning and scheduling in hospitals [1–5]. For example, Decker and Jinjiang propose a multi-agent solution using the Generalized Partial Global Planning (GPGP) approach that preserves the existing human organization and authority structures, while providing better system-level performance (increased hospital unit throughput) [1]. Kutanoglu and Wu investigate a new method based on a distributed and locally autonomous decision structure using the notion of combinatorial auction [2]. Oddi and Cesta explore constraint-based scheduling techniques and implement a mixed-initiative problem solving approach in order to achieve satisfactory solutions to the problem of managing medical resources in a hospital environment [3]. Daknou, Zgaya, Hammadi and Hubert focus on treatment scheduling for patients at emergency department

✉ Fu-Shiung Hsieh
fshsieh@cyut.edu.tw

[1] Department of Computer Science and Information Engineering, Chaoyang University of Technology, Taichung, Taiwan

in hospitals [4] based on multi-agent systems (MAS). Marinagi et al. consider patients as the entities that evaluate hospital services to achieve quality services [18].

Among the research issues mentioned above, how to properly handle patients timely under resource constraints to reduce risk is an essential one. In a hospital, an urgent patient often needs to be handled properly by a time constraint. It is critical to determine whether the medical procedure of a patient can be completed by a time constraint based on the available resources in hospitals. The problem to determine whether a single patient can be handled by a time constraint can be stated as a constraint satisfaction problem (CSP) [9, 17] in existing artificial intelligence (AI) literature whereas the problem to determine whether multiple patients can be handled by a time constraint can be described as a patient scheduling problem. Scheduling patients in a hospital is a challenging problem due to computational complexity involved, distributed resources and dynamic changing environment. Our interest is to propose a viable approach to develop a scalable and sustainable software system for scheduling patients. Given a set of requests from patients in a hospital, the problem is to find schedules that handle patients' requests timely. The objective is to propose a scalable and dynamic scheme to shorten patient stay in hospital under resource constraints and precedence constraints of medical workflows.

In AI, MAS provides a flexible architecture to model operations and dynamically allocate resources in a hospital to handle patients [6, 7]. Entities and resources in a hospital such as doctors, staffs, specialists and nurses can all be modeled as agents. The key issue is how to make these agents work together coherently to handle patients timely. Although MAS provides a flexible architecture to capture the characteristics of entities in medical information systems, how to develop a patient scheduling algorithm that can be applied by individual agents to generate coherent schedules for health care workers to handle patients timely is an important issue.

In this paper we exploit recent advancement in MAS, scheduling theory and information technologies to propose a framework for scheduling patients and develop a scalable, sustainable and flexible solution methodology by extending the preliminary results presented in [33]. To develop a solution methodology, the concept of cooperative distributed problem solving (CDPS) [8], formal temporal models [13, 15], optimization theory [19] and negotiation mechanism [14] are adopted in this study to make agents work together to schedule patients timely. To propose a pragmatic methodology for solving the patient scheduling problem in MAS, a proper architecture and suitable models must be selected to capture the operations, interaction and workflows of agents. In our architecture, the workflow to be performed by an agent is represented by a workflow agent and each resource is modeled by a resource agent. To make these agents work together coherently to handle patients, a negotiation mechanism is used. In MAS, a well known protocol for coordination and negotiation is the contract net protocol (CNP) [14], which defines procedures such as task announcement, bidding mechanism, contract awarding and negotiation [20]. There are a lot of works on distributing tasks in MAS with CNP [21–25]. We focus on how agent technology can be further developed to support collaborative scheduling.

As we use the contract net protocol as the negotiation protocol, messages such as Call for Proposal, Proposals and Request sent and processed by agents must be specified by a structured format that can be easily parsed and interpreted by agents. Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding structured data in a human-readable and machine-readable format. Furthermore, as we develop the proposed system based on Java programming language, there are several application programming interfaces (APIs) available for processing XML data. Therefore, while Ontology is a standard content format supported in FIPA, we adopt Extensible Markup Language (XML) as the format for the messages exchanged between agents. Using XML as a data exchange format also facilitates interactions with other information systems such as HL7.

Our approach is different from these existing ones in that we introduce timed Petri net models in workflow agents and resource agents. Petri net is a graphical formal model for modeling and analysis of workflows [26–28]. The Petri Net Markup Language (PNML) [29] is an XML-based interchange standard for representing Petri nets. Many Petri net tools support PNML format, e.g. WoPeD, Renew, PNK, PEP, VIPtool) [30]. Therefore, PNML is used as the format for representing the Petri net models of agents in the system. To endow each agent with the knowledge and capability to perform operations in workflows, we construct the timed Petri net (TPN) model [15] for each workflow agent and resource agent. Our solution methodology combines MAS architecture with Petri net models. To formulate the optimization problem for a workflow agent, we first obtain the parameters from the corresponding timed Petri net models. Due to the dependency of workflows between agents in MAS, the workflow scheduling problem can be decomposed into a number of interrelated workflow scheduling subproblems that are solved by individual agents. The scheduling algorithm is developed by applying optimization theories based on minimum cost network flow problem formulation [19].

In our system, we define four types of agents, including workflow agents, resource agents, patient agents and scheduling agents. We construct models for workflow agents and resource agents. A medical workflow usually involves a complex process described by a workflow. To

facilitate optimization of patient schedule, we adopt timed Petri nets (TPN) [13] as the modeling tool in this paper instead of using informal workflow specification languages such as XPDL [10], BPMN [11] and WS-BPEL [12]. We propose a scheduling method based on interactions between patient agents, workflow agents and resource agents using contract net protocol (CNP) [14] to efficiently allocate resources. We develop a problem solver based on Java Agent Development Environment (JADE) to verify our method. We illustrate effectiveness and scalability of our approach by examples. To illustrate the benefit of our approach, we compare the performance of our method with a heuristic rule commonly used in practice. In hospitals, the rule of first-come first-serve (FCFS) has been widely adopted and used in practice. An interesting issue is to compare the performance of our approach with that of the FCFS rule for handling patients. In this paper, we will demonstrate the benefit of our approach by examples. In addition, we also analyze and demonstrate scalability of our approach by experiments.

The remainder of this paper is organized as follows. In Section 2, we review related works. In Section 3, we describe the patient scheduling problem and our approach. In Section 4, we introduce the TPN models for agents. In Section 5, we formulate the scheduling problem based on the construction of network models obtained from TPN models and develop solution algorithms for it. We present our prototype system implementation and examples in Section 6. In Section 7, we analyze scalability of our approach and present the results. We conclude this paper in Section 8.

2 Related works

In practice, hospitals and healthcare service providers usually use a clinical guideline to guide decisions regarding diagnosis, management, and treatment in a specific of healthcare area [37, 38, 41]. In the existing literature, Fox et al. pointed out that existing clinical guideline suffer from several drawbacks such as inaccessibility of guidance at the point of care, ambiguity and inapplicability in local settings [39]. They turned their attention from clinical guidelines to decision support services and developed tools to author, edit, publish and maintain process for active decision support and guideline services in the OpenClinical project (www.openclinical.net). In [40], Kaiser and Marcos found that modeling of clinical practice guidelines is a labor-intensive task even with tool assistance. They facilitated the clinical practice guidelines modeling task based on a set of procedural patterns described in an implementation-independent notation in BPMN 2.0 that can be then semi-automatically transformed into one of the alternative executable clinical practice guideline languages. However, BPMN 2.0 lack formal analysis method and it is not a suitable model for analyzing, formulating and solving scheduling problems. Therefore, we adopt Petri nets as the tools to model workflows and schedule patients in hospitals. A wide variety of free open source editing software tools are available. Users only need to create graphical workflow and resource models in Petri net for entities in hospitals and specify the relevant data in our approach. Our model transformation component will formulate and solve the scheduling problem behind the scenes.

The essence of our model transformation approach is inspired by the concept of model driven development (MDD) [47, 48], which has been widely adopted in software development by expressing models less bound to the underlying implementation technology and closer to the problem domain. The benefits of MDD include reducing the barrier and learning time for users by using graphical standardized platform independent models to enhance efficiency of communication and accommodate changes. In our approach, we use Petri nets as the platform independent models to specify the models for the scheduling problem. As our approach is based on MDD, our approach enjoys some of the benefits of the MDD approach. In practice, workers of hospitals are usually not familiar with optimization theory. Application of optimization theory to patient scheduling problem requires some background on the mathematical modeling for the problem. Therefore, it is very difficult for workers of hospitals to apply software tools that are based on optimization theory. Our approach reduces the barrier and learning time for users not familiar with optimization theory by using a graphical workflow model based on Petri nets.

The patient scheduling problem considered in this paper and our approach to scheduling patients are different from those reported in [16, 34–36]. The appointment scheduling of outpatient surgeries in a multistage operating room department with stochastic service times serving multiple patient types has been addressed in [34]. In [35], the authors formulated and solved a discounted infinite-horizon Markov decision process for scheduling cancer treatments in radiation therapy units. The goal of [36] is to propose and solve a new formulation of the recently-formalized patient admission scheduling problem by a meta-heuristic approach. In [16], the author proposed a hierarchical goal programming (HGP) model for scheduling the outpatient clinics. This paper is also different from our previous works [31]. Paper [31] demonstrates the capabilities of PNML in the development of context-aware applications instead of an industrial interchange format only. However, Papers [31] does not address the scheduling problem in MAS.

There are many papers and several frameworks that apply agent technology in hospitals or healthcare systems. For

example, Agent.Hospital [43, 44] is an open agent-based framework for distributed applications in the healthcare domain. A review of the most recent literature of applications of agents in healthcare is discussed in [42], which indicates a growing interest of researchers in the development of agent based applications in healthcare systems. This paper is also based on agent technology. Each agent in our system is an autonomous, co-operative and intelligent entity able to collaborate with other agents in handling patients. This paper is different from papers [45, 46], which also use agents in scheduling hospitals. The problem studied in this paper is to shorten patient stay in a hospital and is different from the problem setting of [45], which presents an agent based model for the selection of optimal mix for patient admissions in hospitals. In [46], patients and hospital resources are modeled as agents with individual goals that negotiate to find appropriate solutions without applying optimization theory. This paper is different from [46] in that we combine the distributed architecture of multi-agent systems with optimization theory to attain the benefits of improving performance in distributed environment with decision autonomy and flexibility of agents.

3 Patient scheduling problem and the proposed approach

Patients in a hospital are handled by following medical procedures. A medical procedure abstracts the workflows and operations in a hospital. A typical medical procedure may consist of a number of steps such as registration, diagnosis, radiology test, blood examination, anesthesia, surgery, intensive and discharge, etc. Different steps throughout the lifecycle of a medical procedure usually require distinct resources for processing. In practice, these steps may be performed by different hospital workers or resources such as doctor, staff, specialist and nurse. Due to the complexity of the operations performed and the distributed information and resources, medical workflows in a hospital require considerable coordination of resources. How to effectively manage the concurrent workflows in the system is a challenge. We are mainly concerned with the management of related resources for effective enactment of these steps. The patient scheduling problem is to find schedules that allocate resources over the scheduling horizon to minimize earliness/lateness in handling patients while satisfying all the precedence constraints and resource constraints associated with the workflows. In this paper, we will propose a scheduling system based on MAS to determine the schedules to handle patients timely based on the available resources in a hospital. In this section, we focus on the architecture for solving patient scheduling problem. Formal problem formulation will be described in Section 4 based on the models of medical workflows and activities described in Section 3.

Our approach to scheduling patients is based on collaboration of agents in the system. Figure 1(a) shows our architecture for solving the patient scheduling problem. There are four types of agents, including patient agents, medical workflow agents, resource agents and scheduling agents. A patient is represented by an agent called patient agent. Doctors, staffs, specialists and nurses and other workers in a hospital are modeled by resource agents. The medical procedure for a patient usually consists of a sequence of medical workflow agents. A scheduling agent implements the algorithm to optimize schedules. The last medical workflow agent in a medical procedure is called the target medical workflow agent. The request from a patient agent propagates from the patient agent to its target medical workflow agent progressively until it reaches the first medical workflow agent at the upstream in the medical procedure. For example, consider a medical procedure with three medical workflow agents: radiology agent, anesthesia agent and surgery agent. For this example, the target medical workflow agent for the patient agent is the surgery agent. Therefore, the scheduling request will propagate from the patient agent to the surgery agent and then the anesthesia agent and finally the radiology test agent. Each medical workflow agent applies a scheduling algorithm by interacting with the scheduling agent and relevant resource agents. If a schedule can be found, the medical workflow agent will issue a request to its upstream medical workflow agent next. The processes will continue until the scheduling request reaches the first medical workflow agent at the upstream in the medical procedure. Figure 1(b) illustrates the scheduling requests and responses sent between a patient agent and the relevant upstream medical workflow agents.

The interaction between a patient, the relevant medical workflow agents, resource agents and scheduling agent can be specified by a sequence diagram in Unified Modeling Language (UML). Let PA be a patient agent. Let MA1, MA2 and MA3 denote three medical workflow agents, including surgery agent, anesthesia agent and radiology test agent, in the medical procedure and RA1, RA2 and RA3 be three different resource agents that can perform the operations of MA1, MA2 and MA3, respectively. Figure 2(a) shows the UML sequence diagram of PA, MA1, MA2, MA3, RA1, RA2 and RA3.

As the function of patient agents is to provide graphical user interface (GUI) for users to specify and issue the scheduling requests, we only briefly describe the internal structure of medical workflow agents, resource agents and scheduling agents in Fig. 2(b), (c) and (d), respectively. With the exception of scheduling agents, each agent provides a GUI for users to interact with other agents in the system. The service discovery module facilitates the processes to discover

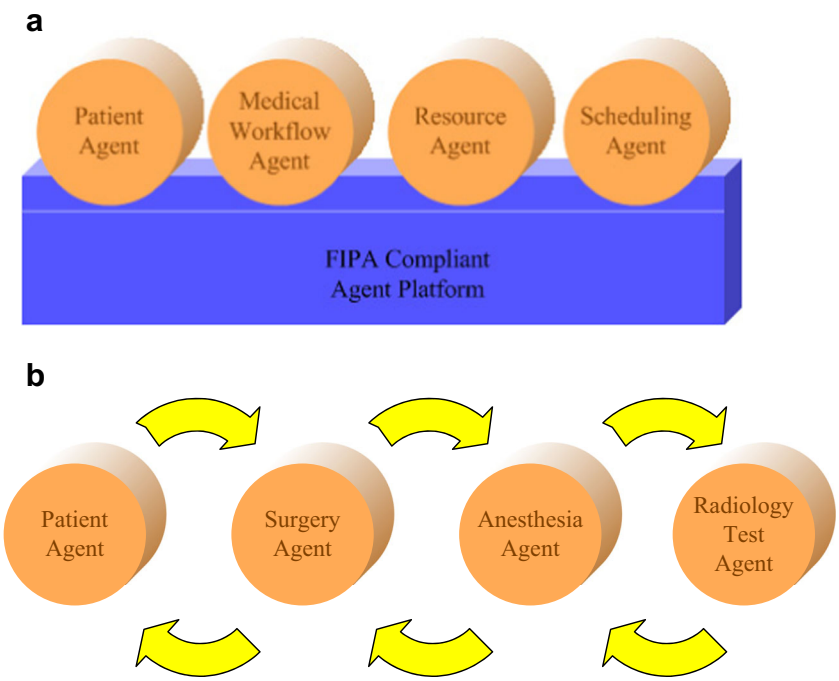


Fig. 1 (**a**) Architecture to schedule medical workflows in health care systems. (**b**) Scheduling requests and responses sent between a patient agent and the relevant upstream medical workflow agents

the services provided by other agents. A medical workflow agent consists of a GUI to describe relevant workflow information. Figure 2(b) shows the internal structure of a medical workflow agent. The properties of a medical workflow agent are described by an XML file (Workflow.xml). Table 7 in Appendix I shows an example of Workflow.xml used in our system. The medical workflow model is described by a timed Petri net model., which will be described in the nest section. A medical workflow agent needs to discover and interact with other agents. A request from a patient agent is represented in XML format. Table 8 in Appendix I shows a patient request example in our system. On receiving a request from a patient agent, a medical workflow agent will parse the patient request to extract the medical workflow type and the deadline for handling the patient.

Figure 2(c) shows the internal structure of a resource agent. The capability of a resource agent is defined by a GUI and is described by an XML (Resource.xml) file. Table 9 in Appendix I shows an example of Resource.xml used in our system. The activities in the medical workflow to be performed by resource agents are also represented by timed Petri net models, which will be defined in the next section. Figure 2(d) shows the internal structure of a scheduling agent. A scheduling agent is invoked by a medical workflow agent when a patient request is received. Based on timed Petri net models, a scheduling agent constructs a network model and combines it with a subgradient algorithm to optimize the schedules. The timed Petri net models will be defined in Section 4 and the network model will be introduced in Section 5.

4 TPN models for medical workflows and activities

In a hospital, operation of one worker may depend on or influence the operations of other ones. The scheduling problem is dynamic and subject to several constraints. To deal with the dynamic scheduling problems in a hospital, one strategy is to develop a software agent that elicits the scheduling problem based on well-known process specification languages or models to automate the scheduling problem formulation. In this way, users do not need to get acquainted with mathematical optimization theories to use our tools to schedule operations. Petri nets have been widely adopted as a tool for modeling processes in industry [15]. The medical procedures in a hospital usually form complex workflows which may involve synchronous/asynchronous/concurrent workflows and activities that can be modeled by Petri nets. Therefore, we adopt timed Petri net (TPN) [15] in this paper to model the medical workflows and activities performed by agents and use PNML format to represent the model.

A TPN [15] G is a five-tuple $G = (P, T, F, \mu, m_0)$, where P is a finite set of places, T is a finite set of transitions, $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation, $\mu : T \to Z^+$) is a mapping that specifies the discrete firing time for each transition, where Z^+ is the set of nonnegative integer numbers, and $m_0 : P \to Z^{|P|}$ is the initial marking of the PN with Z as the set of nonnegative integers. A Petri net with initial marking m_0 is denoted by $G(m_0)$. A marking of G is a vector $m \in Z^{|P|}$ that indicates the number of tokens in each place under a state. ${}^\bullet t$ denotes the set of

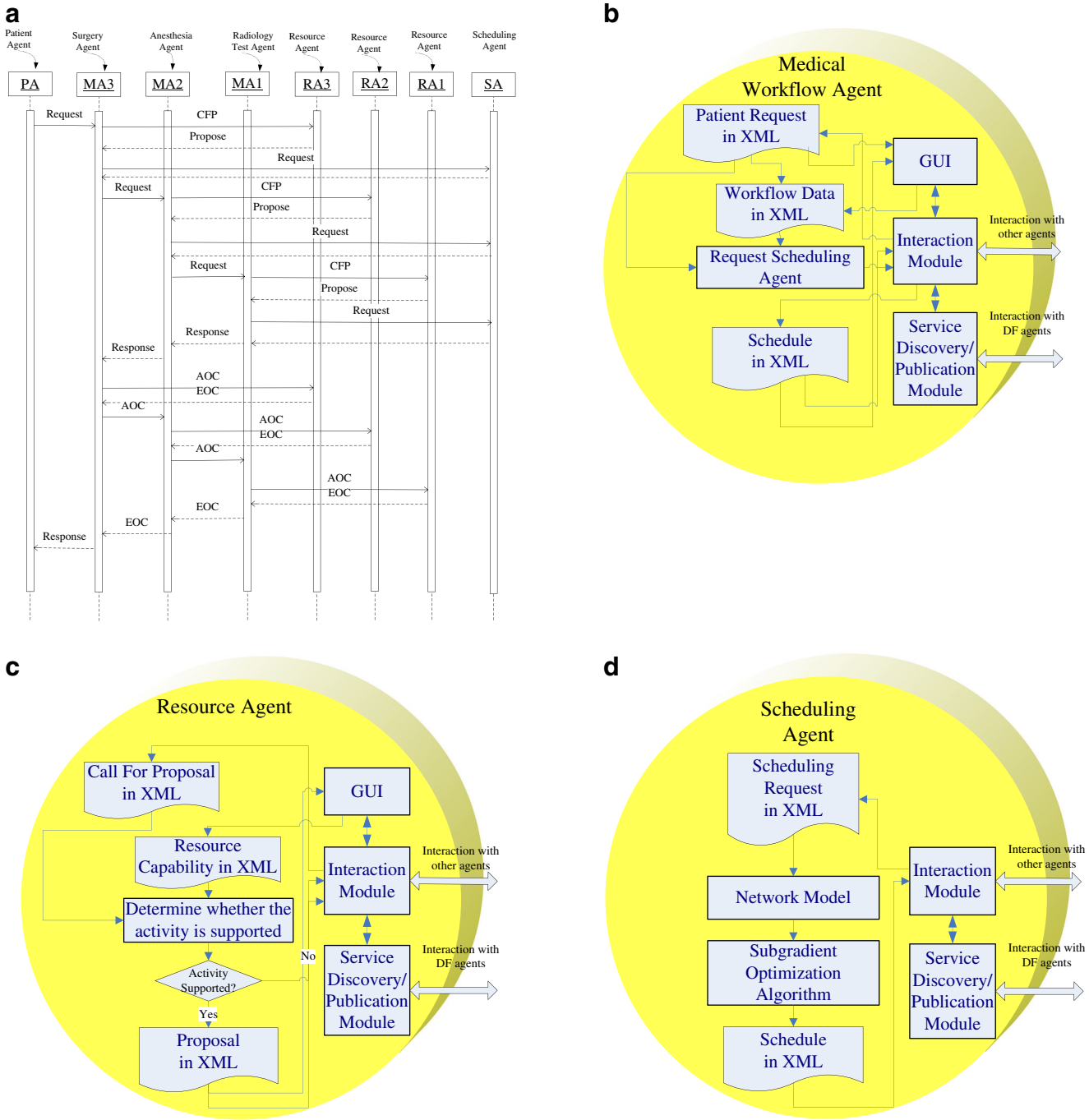


Fig. 2 (**a**) A sequence diagram for scheduling patients based on collaboration of agents CFP: Call for proposals AOC: Awarding of contracts. EOC: Establishment of contracts. (**b**) Internal structure of a medical workflow agent. (**c**) Internal structure of a resource agent. (**d**) Internal structure of a scheduling agent

input places of transition t. A transition$t$ is enabled and can be fired under m iff $m(p) \geq F(p,t) \forall p \in^{\bullet} t$. ${}^{\bullet}p$ denotes the set of input transitions of place p and $p^{\bullet}$ denotes the set of output transitions of place p. In a TPN, each transition is associated with a firing time. Firing a transition removes one token from each of its input places and adds one token to each of its output places. A marking m' is reachable from m if there exists a firing sequence s bringing m to m'. A TPN $G = (P, T, F, \mu, m_0)$ is live if, no matter what marking has been reached from m_0, it is possible to ultimately fire any transition of G by progressing through some further firing sequence.

To model a medical workflow as a TPN, we use a place to denote a state in the workflow while a transition denotes

an operation. A Petri net is called a marked graph [15] if it is an ordinary Petri net such that each place p has exactly one input transition and exactly one output transition, i.e., $|{}^\bullet p| = |p^\bullet| = 1$ for all $p \in P$. A medical workflow is modeled by a subclass of Petri nets that is very similar to marked graphs. We call this subclass class of Petri nets an acyclic sequential timed marked graph (ASTMG) as follows.

Definition 4.1 A timed marked graph $TSMGW'_n = (P'_n, T'_n, F'_n, \mu'_n, m'_{n0})$ is called a sequential timed marked graph if each transition in T'_n has only one input place and one output place.

Definition 4.2 A transition in T'_n without any input place is called a terminal input transition. A transition in T'_n without output place is called a terminal output transition.

Definition 4.3 An acyclic sequential timed marked graph (ASTMG) $W_n = (P_n, T_n, F_n, \mu_n, m_{n0})$ is obtained by augmenting $W'_n = (P'_n, T'_n, F'_n, \mu'_n, m'_{n0})$ with an input place ε_n and a service output place θ_n and adding an arc connecting ε_n to the terminal input transition of W'_n and adding an arc connecting the terminal output transition to θ_n.

Definition 4.4 The model of a medical workflow agent w_n is an ASTMG $W_n = (P_n, T_n, F_n, \mu_n, m_{n0})$.

Figure 3(a) shows the Petri net models of three medical workflows w_1 through w_3. Note that the individual medical workflow model for each step has one start state, several processing state and one finished state. For example, the workflow Petri net model w_1 for Step 1 consists of one start state (p_1), one busy state (p_2) and one finished state (p_3).

To handle a patient, the Petri net models of multiple workflow agents can be merged to construct a complete medical workflow We define the operator "$\|$" to merge multiple Petri nets through service input place and service output place or common places, transitions, or arcs.

Definition 4.5 Given two Petri nets $G_1 = (P_1, T_1, W_1, m_{10})$ and $G_2 = (P_2, T_2, W_2, m_{20})$, $G_1 \| G_2 = (P, T, W, m_0)$, where $P = P_1 \cup P_2$, $T = T_1 \cup T_2$, $W(p,t) = \begin{cases} W_1(p,t) \; if \; p \in P_1 \; and \; t \in T_1 \\ W_2(p,t) \; if \; p \in P_2 \; and \; t \in T_2 \end{cases}$, $W(t,p) = \begin{cases} W_1(t,p) \; if \; p \in P_1 \; and \; t \in T_1 \\ W_2(t,p) \; if \; p \in P_2 \; and \; t \in T_2 \end{cases}$, and $m_0(p) = \begin{cases} m_{10}(p) \; if \; p \in P_1 \\ m_{20}(p) \; if \; p \in P_2 \end{cases}$.

Definition 4.6 Let S denote the set of workflow agents required to construct a complete medical workflow. $CW = \|_{n \in S} W_n$.

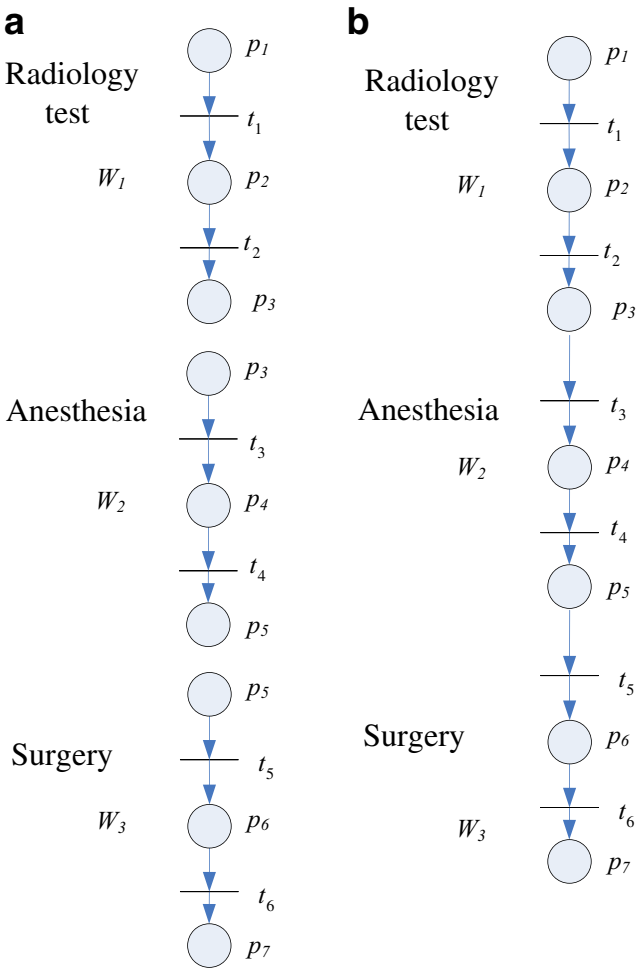


Fig. 3 (**a**) ASTMG models of workflow agents $w_1 \sim w_3$. (**b**) A medical workflow model obtained by merging the models of agents $w_1 \sim w_3$

By applying the operator "$\|$" to the workflow models of $w_1 \sim w_3$, we can construct a complete medical workflow in Fig. 3(b).

An activity is a sequence of operations to be performed by a certain type of resources. The Petri net model for the $k-th$ activity of resource r is described by a Petri net Γ^k_r that starts and ends with the resource idle state place p_r as follows.

Definition 4.7 Petri net $\Gamma^k_r(m^k_r) = (P^k_r, T^k_r, F^k_r, \mu^k_r, m^k_r)$ denotes the $k-th$ activity of resource r. There is no common transition between Γ^k_r and $\Gamma^{k'}_r$ for $k \neq k'$. The initial marking $m^k_r(p_r) = 1$ and $m^k_r(p) = 0$ for each $P^k_r \setminus \{p_r\}$, where p_r is the idle state place of resource r.

Figure 4 illustrates a resource activity associated with the work flow $w_1$in Fig. 3(a). To solve the dynamic scheduling problem, a problem formulation based on the proposed Petri net models will be detailed in the next section.

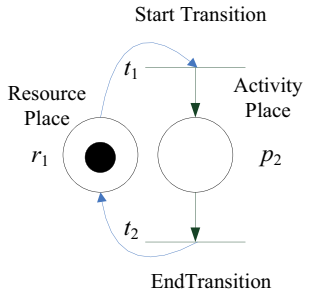


Fig. 4 Specification of a resource activity by a Petri net

5 Scheduling problem formulation and our solution approach

Although Petri net models can be used to represent the workflows and activities in medical procedures, it is hard to optimize the schedules using Petri nets. In our approach, we develop a method to transform the proposed discrete timed Petri nets to a network that can capture the flow of tokens (representing patients and resources) at different time points in different states. We formulate and solve the patient scheduling problem based on the network constructed. To formulate the optimization problem, we first obtain the parameters from the corresponding discrete timed Petri net models. The constraints of the optimization problem are then represented by a network to facilitate the development of our solution algorithm. Formally, the scheduling problem for each workflow agent is formulated based on the following notation.

$\boldsymbol{WA} = \{1, 2, 3, \ldots, N\}$: The set of workflow agents in the system.

$\mathrm{N} = |\boldsymbol{WA}|$: The number of workflow agents in the system.

$\boldsymbol{PA} = \{1, 2, 3, \ldots, \Pi\}$: The set of patient agents in the system.

Π: $\Pi = |\boldsymbol{PA}|$, the number of patient agents.

W_n : The workflow model of workflow agent $n \in \boldsymbol{WA}$.

$\mathrm{K_n}$: The number of different resource activities in W_n.

Γ_r^k : The k-th resource activity in W_n, $k \in \{1, 2, \ldots, \mathrm{K_n}\}$. Suppose the k-th resource activity in W_n is performed by resource agent r_k.

π_{nk} : The processing time of the k-th resource activity Γ_r^k in W_n with $\pi_{nk} = \mu_r^k(t_s^k) + \mu_r^k(t_e^k)$, where t_s^k and t_e^k denote the starting and ending transitions of the k-th activity of resource agent a_r.

τ_p : The due date of patient agent $p \in \boldsymbol{PA}$.

d_p : The number of patients in the request of a patient agent p.

T : The total number of time periods.

t : A time period and $t \in \{1, 2, 3, \ldots, T\}$.

R_{rt} : The capacity of resource r at time period t, where $R_{rt} = m_{r0}^k(r)$.

v_{pkt} : The number of patients in the request of patient agent p that requires resource r_k for processing the k-th resource activity in W_n during time period t with $v_{pkt} \geq 0$ and $v_{pkt} \in Z^+$, the set of non-negative integers.

η_{pt} : The number of patients in workflow W_n finished during time period t; i.e., $\eta_{pt} = v_{pK_n(t-\pi_{nK_n})}, \forall p, \forall t$.

h_{pkt} : The number of patients in the request of patient agent p waiting at for the k-th resource activity in W_n at the beginning of period t, where $h_{pkt} \geq 0$ and $h_{pkt} \in Z^+$, the set of non-negative integers.

Without loss of generality, we assume different resource activities in a workflow W_n are performed by different resources. For the k-th resource activity in W_n, the time that elapses from a start node to the corresponding end node is equal to the processing time π_{nk} of the k-th resource activity.

Note that the due date τ_p for workflow agent w_n is either set by its corresponding patient agent or its downstream workflow agents in the negotiation processes.

To timely handle a patient, we introduce an earliness/lateness penalty function $\psi(t, \tau_p)$ for patient agent $p \in \boldsymbol{PA}$ completed at time t. Note that the earliness/lateness penalty function $\psi(\tau_p, \tau_p)$ satisfies the following requirements:

(i) $\psi(\tau_p, \tau_p) = 0$ and (ii) $\psi(t, \tau_p) \geq 0$ to handle patients just-in-time.

The patient scheduling problem is to find the schedules to allocate resource over the scheduling horizon to minimize earliness/lateness in handling patients while satisfying all the precedence constraints and resource constraints. Given a set $\boldsymbol{PA}$ of patient agents with specified requirements and a set $\boldsymbol{WA}$ of workflow agents, the patient scheduling problem is formulated as follows.

Patient Scheduling Problem (PSP_n) for Medical Workflow Agent n with model W_n

$$\min \sum_{p=1}^{\Pi} \sum_{t=1}^{T} \psi(t, \tau_p) \eta_{pt}$$

$$s.t.$$

$$h_{p11} = d_p \ \forall p \tag{5.1}$$

$$h_{p1(t+1)} = h_{p1t} - v_{p1t} \ \forall p, \forall t \tag{5.2}$$

$$h_{pk(t+1)} = h_{pkt} - v_{pkt} + v_{p(k-1)(t-\pi_{n(k-1)})} \tag{5.3}$$

$$\forall p, \forall t, k \in \{2, 3, \ldots, \mathrm{K_n}\}$$

$$h_{p(K_n+1)(t+1)} = h_{p(K_n+1)(t+1)} + v_{pK_n(t-\pi_{nk})} \forall p, \forall t \tag{5.4}$$

$$\eta_{pt} = v_{pK_n(t-\pi_{nK_n})} \ \forall p, \forall t \tag{5.5}$$

$$\sum_{p=1}^{\Pi} \sum_{\tau=t-\pi_{nk}+1}^{t} v_{pk\tau} \leq R_{r_k t} \qquad \forall k \in \{1, 2, \ldots, \mathrm{K_n}\}, \forall t \tag{5.6}$$

To ensure the allocated time slots cannot exceed the overall capacities of resources, the capacity constraints defined in inequality (5.6) must be satisfied. Furthermore, the flow balance (5.1), (5.2), (5.3) and (5.4) must be satisfied.

In this paper, we combine optimization theories with multi-agent system architecture to allocate resources and perform operations of workflows. We adopt a divide and conquer approach to achieve optimization locally. Optimization is achieved by applying the Lagrangian relaxation technique to develop a solution algorithm for the dual problem of the problem PSP_n. In problem PSP_n, we observe that the coupling among workflows of different patients is caused by contention for resources. Based on this observation, we apply Lagrangian relaxation to relax resource capacity constraints (5.6) and form the Lagrangian function as follows:

$$L(\lambda) \equiv \sum_{p=1}^{\Pi} \min \sum_{t-1}^{T} (\psi(t, \tau_p)\eta_{pt}) + \sum_{k=1}^{K_n} \sum_{t=1}^{T} \lambda_{kt} \left(\sum_{\tau=t-\pi_{nk}+1}^{t} v_{pk\tau} - R_{r_k t} \right) \quad (5.7)$$

s.t. constraints (5.1)∼(5.5), where λ_{kt} is the associated Lagrange multiplier that must be nonnegative.

Note that the flow balance equations described by constraints (5.1)∼(5.5) can be represented by a network flow model, which is constructed by applying our network construction algorithm.

To represent the timing of an activity, we transform the discrete timed Petri net model of a medical workflow agent to a network model. We first create a time axis. We use a sequence of nodes placed horizontally in the time axis to denote different time points of a state. To represent a token entering a start state, we create a sequence of start nodes along the time axis. To represent a token leaving an end state, we create a sequence of end nodes along the time axis. To represent the flow of tokens from one start transition to the corresponding end transition, we create a sequence of arcs connecting start nodes to end nodes. In addition, we use one source node and one sink node. In this way, the flow of token at different time periods can be represented by a network model. To penalize earliness/lateness, we assign penalty cost on the arcs that connect the end nodes to the sink node. We assign the cost of ω_{pt} for patient p to minimize earliness/lateness. The algorithm to construct the network associated with an activity is as follows. By applying the minimum cost flow algorithm [19], the flow will follow the minimal cost path in the network.

Algorithm 1 Network Construction Algorithm

Input: W_n, Γ_r^k, T
Output: Ψ_n
Step 1: Create a set S of T start nodes for each start transition t_s
Step 2: Create a set E of T end nodes for each end transition t_e
Step 3: For each start node in $s_t \in S$
  If $t + \mu \leq T$
    Create an arc a connecting s_t to the end node $e_{t+\mu}$
    $A \leftarrow A \cup \{a\}$
  End If
End For
Step 4: For each arc$a \in A$
  Set arc cost to the arc a associated with $v_{pk\tau}$ according to the coefficient of $v_{pk\tau}$ in (5.7)
End For
Step 5: For each arc $a \in A$
  Set arc capacity to the arc a associated with $v_{pk\tau}$ according to $m_{r0}^k(r)$
  End For

To optimize the schedule, the cost of each arc should be assigned properly. Each arc in the network will be assigned some cost properly according to Lagrangian relaxation technique described later in this section.

The dual problem corresponding to the problem PSP_n is defined as follows:

$$\max_{\lambda \geq 0} L(\lambda)$$

A subgradient algorithm is used in this paper to solve the above problem. Subgradient method is an iterative method for solving convex optimization problems. It can be used with a non-differentiable objective function. When the objective function is differentiable, subgradient method for unconstrained problems uses the same search direction as the method of steepest descent. Moreover, by combining the subgradient method with primal or dual decomposition techniques, it is possible to develop a distributed algorithm for a problem. The optimal Lagrange multiplier is determined by solving the dual problem $\max_{\lambda \geq 0} L(\lambda)$. Since $L(\lambda)$ is not differentiable due to integrality constraints in subproblems, we adopt a simple, iterative subgradient method [32] to solve the dual problem. Our approach to finding a solution of $\max_{\lambda \geq 0} L(\lambda)$ is based on an iterative scheme for adjusting Lagrangian multipliers according to the solutions of minimum cost flow subproblems.

Let i be the iteration index. Let v^i denote the optimal solution to minimum cost flow subproblems for given

Lagrange multipliers λ^i at iteration i. We define the subgradients of $L(\lambda)$ with respect to Lagrangian multipliers λ^i as follows:

$g^i_{kt} = \sum_{p=1}^{\Pi} \sum_{\tau=t-\pi_{nr}+1}^{t} v^i_{pk\tau} - R_{r_k t}, \forall k = 1, \ldots, K_n, \forall t = 1, \ldots, T$ The subgradient method proposed by Polyak [32] is adopted to update λ as follows: $\lambda^{i+1}_{kt} = \begin{cases} \lambda^i_{kt} + \alpha^i g^i_{kt}, & \text{if } \lambda^i_{kt} > 0 \text{ or if } \lambda^i_{kt} = 0 \text{ and } g^i_{kt} \geq 0 \\ 0, & \text{if } \lambda^i_{kt} = 0 \text{ and } g^i_{kt} < 0 \end{cases}$,

where $\alpha^i = \dfrac{\beta[\bar{L}(\lambda^*) - L(\lambda^i)]}{\sum_{k,t} (g^i_{kt})^2}$ and $\bar{L}$ is an estimate of the optimal dual cost and $0 < \beta < 2$.

Figure 5 shows the flow chart of our algorithm. Iterative application of the subgradient algorithm will converge to an optimal dual solution (v^*, λ^*). It should be emphasized that Lagrangian relaxation does not guarantee the optimal solution to the underlying problem. Rather, it finds an optimal solution to a *relaxation* of it. Furthermore, while Lagrangian relaxation will yield the optimal objective function value for the *linear relaxation* of the underlying integer program, it is not guaranteed to produce a feasible solution. Thus the solution generated may not satisfy the complementary slackness conditions. In case the solution is not feasible, we must develop a heuristic algorithm to find a feasible solution. In our system, we implement a simple heuristic algorithm that removes the excessive flows from the arcs with capacity violation by setting the arc capacity to zero and reroute these flows to other part of the network based on minimum cost flow algorithm.

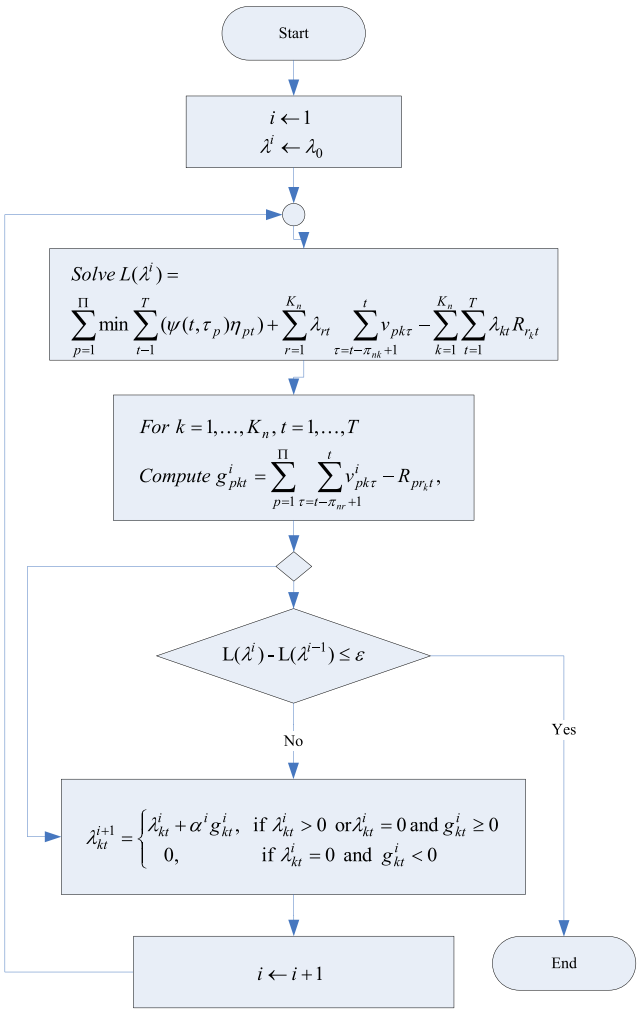


Fig. 5 Flow chart of our algorithm

6 Numerical results

We have developed a multi-agent patient scheduling system based on JADE and the methodology proposed in previous sections. In this section, we will first briefly introduce our implementation and illustrate the functions of our system. We then illustrate the effectiveness of our method by examples.

6.1 The Scheduling system

In our system, each resource in a hospital, including doctors, specialists and nurses, is modeled by a resource agent. The patients to be handled in a hospital are represented by patient agents. Our approach to schedule patients relies on interaction between patient agents, medical workflow agents, resource agents and scheduling agents. Interactions among different types of agents are based on a negotiation mechanism that extends the well-known contract net protocol (CNP) [14]. CNP relies on an infrastructure for individual agents to publish and discover their services and communicate with each other based on the ACL language defined by the FIPA international standard for agent interoperability. JADE is a middleware that facilitates the development of multi-agent systems. Each type of entities in the hospital is implemented as an agent in JADE platform. Messages exchanged by JADE agents have a format specified by the ACL language defined by the FIPA international standard for agent interoperability. One of the essential functions required for agents in the system to discover each others is directory service. To apply the CNP, an agent must be able to search for the services provided by the other agent(s). In our scheduling system, we define different service types for patient agents, workflow agents and resource agents. Each time an agent is added to the system, its services are published through the DF (Directory Facilitator) agent in JADE.

Patient agents, workflow agents and resource agents are accompanied with proper graphical user interface (GUI) for users to input and display the data. Scheduling agents, which are invoked by workflow agents, have no GUI as

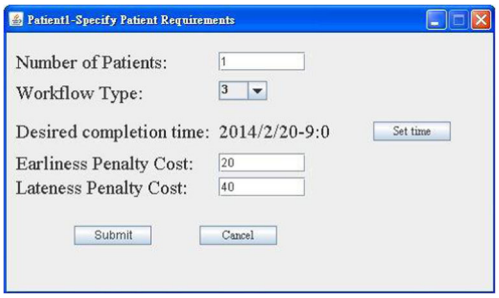


Fig. 6 The GUI for setting the requirement of a patient

they work behind the scenes. Some of the screen shots of our scheduling system are shown in this section. Figure 6 illustrates the GUI for setting the requirement of a patient. Interactions of agents to schedule patients are shown in Fig. 7. The output of our scheduling software includes the schedules for executing each medical workflow and the schedules for performing the operations in medical workflows by each resource agents. Our system provides several forms of outputs to represent the solutions of a patient scheduling problem, including graphs that representing the contracts established between a resource agent for handling patients and the Gantt chart for the schedule of a resource agent. These outputs will be illustrated by an example in the next subsection.

Our system provides several forms of outputs to represent the solutions of a patient scheduling problem. Figure 8 shows the schedule for a resource agent to handle different patients' workflows. The Gantt chart for representing the schedule of a resource agent is shown in Fig. 9. Note that all

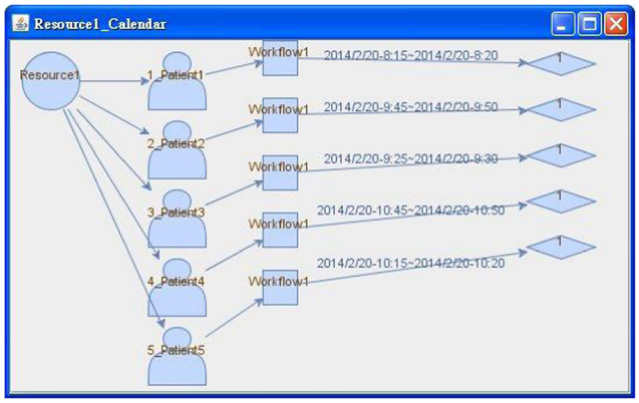


Fig. 8 The schedule for a resource agent to handle different patients' workflows

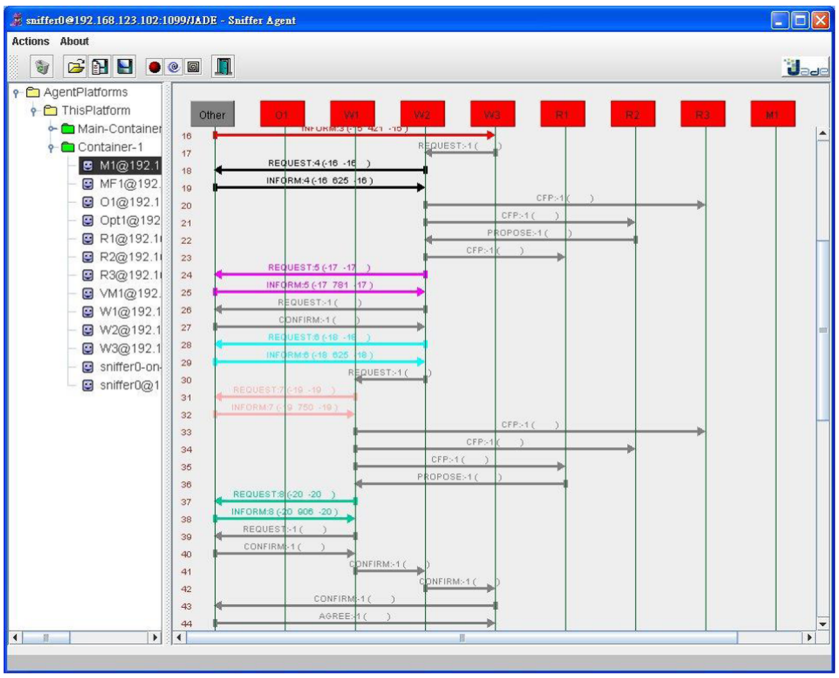
the schedules generated by our system satisfy the capacity constraints of resources and the flow balance constraints of workflows.

To demonstrate the effectiveness of our method, we compare the schedules generated by our system with the ones generated by the rule of FCFS.

6.2 Examples

In the remainder of this section, we use an application scenario to demonstrate the practicality of our scheduling system. Suppose the requests of ten patients are received. The input data for this example are shown in Tables 1 and 2. Table 1 shows three medical workflow agents defined for this example. Each medical workflow represents a task

Fig. 7 Interaction of agents to schedule patients



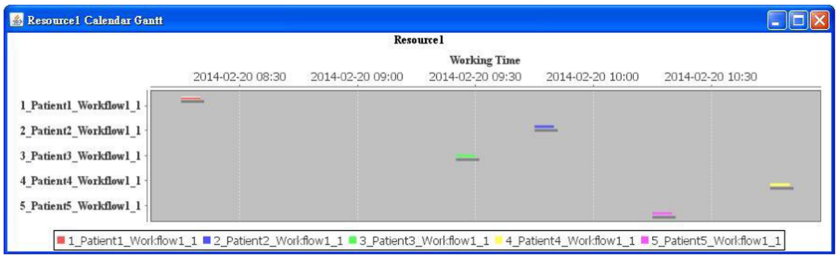


Fig. 9 The Gantt chart for representing the schedule of a resource agent

required for handling a patient. These three medical workflows are distributed in different departments. The three medical workflow models are shown in Fig. 10. A medical workflow is described by a Petri net model represented in PNML file format. There are three types of resource agents. The activities that can be performed by the three types of resource agents are specified by Petri net models in PNML format. The properties of resource agents are listed in Table 2.

The requirements of a patient agent are specified by a due date, a medical workflow type, number of patients, and the penalty cost (earliness penalty cost and lateness penalty cost). In Taiwan, many hospitals provide on-line appointment systems for patients. These systems generate planned appointment time for each patient based on the appointment records of existing patients. Each patient then goes to the hospital according to the planned appointment time. However, the planned appointment time generated by the on-line appointment system may not be accurate. In addition, some patients may arrive at the hospital much earlier than their planned appointment time whereas other patients may arrive at the hospital only a few minutes before the planned appointment time. There exist discrepancy between the planned appointment time and the actual time the medical procedure of a patient is completed.

Table 3 shows the planned appointment time and expected finished time for ten patients. The target medical workflow type for the fourth, the fifth, the sixth and the seventh patients is type-3 medical workflow whereas the target workflow type for all other patients is type-2 workflow. Each type of medical workflows needs to be processed by one resource agent. Table 4 shows the earliness/lateness penalty coefficients for each patient. Based on this application scenario, we compare the performance of our method with the rule of first-come first-serve (FCFS) widely used in practice. The patients may not arrive at the hospital punctually at the planned appointment time. Some arrive earlier whereas others arrive later. Table 5 shows patients' arrival time.

By applying our algorithm, the results are shown in Tables 10, 11, 12, 13, 14, 15, 16, 17, 18, 19 in Appendix II. By applying FCFS to the above example, the results are shown in Tables 20, 21, 22, 23, 24, 25, 26, 27, 28, 29 in Appendix II. Table 6 summarizes the difference in total processing time for this example. The schedules generated by applying the rule of FCFS lead to significant delay for most patients. The results indicate that the discrepancy between the scheduled appointment time and arrival time can be significantly reduced by applying our method. Our method improves accuracy in scheduling patients based on just-in-time philosophy. It indicates that our method outperforms FCFS for this example.

In addition to the examples above, we also conduct experiments for several examples with 20 to 100 patients to study the effectiveness of our approach. Some of the results are shown in Fig. 11, where the average time for handling patients is compared.

7 Scalability analysis and numerical results

To study scalability of the proposed approach, we analyze and conduct experiments to estimate and illustrate how the response time grows as the number of patients and the number of resource activities increase. To illustrate the benefit of our agent based approach, we also analyze and compare the response time of our approach with that of a centralized approach. In this section, we first analyze the response time for our agent based approach. Then we analyze the response time for the centralized approach. Finally, we verify our analysis by numerical results.

Table 1 Definition of Workflows

Workflow id	Medical workflow type	Input type	Output type	PNML file
W1	Diagnosis		1	W1.pnml
W2	Surgery	1	2	W2.pnml
W3	Post Surgery	2	3	W3.pnml

Table 2 Definition of Resource Activities

Resource id	Activity id	Start Transition (Firing time)	End Transition (Firing time)	PNML file
R1	1	$t_1(2)$	$t_2(3)$	R1.pnml
R2	2	$t_3(10)$	$t_4(10)$	R2.pnml
R3	3	$t_5(10)$	$t_6(10)$	R3.pnml

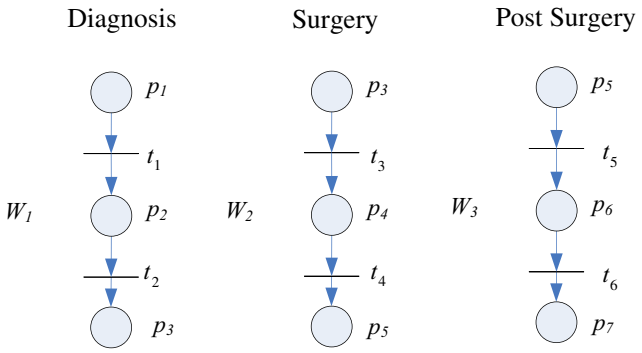


Fig. 10 Three workflow models

For our agent based approach, note that the Lagrangian function for workflow agent w_n is defined by

$$L(\lambda) \equiv \sum_{p=1}^{\Pi} \min \sum_{t-1}^{T} (\psi(t, \tau_p)\eta_{pt}) + \sum_{r=1}^{K_n} \sum_{t=1}^{T} \lambda_{kt} \left(\sum_{\tau=t-\pi_{nk}+1}^{t} v_{pk\tau} - R_{r_kt} \right)$$

s.t. constraints (5.1)∼(5.5) , where λ_{kt} is the associated Lagrange multiplier that must be nonnegative.

To compute $L(\lambda)$ for given λ, it is necessary to solve the minimal cost network flow problem. The computational complexity to solve a minimal cost network flow problem with n nodes and flow of f is $O(n^2 f)$. As the number of nodes in the network associated with $L(\lambda)$ is proportional to $\mathrm{K_n}T$ and the flow is d_p, the computational complexity is $O(\mathrm{K_n^2}T^2 d_p)$.

Note that the Lagrange multipliers are updated as follows: $\lambda_{kt}^{i+1} = \begin{cases} \lambda_{kt}^i + \alpha^i g_{kt}^i, & \text{if } \lambda_{kt}^i > 0 \text{ or if } \lambda_{kt}^i = 0 \text{ and } g_{kt}^i \geq 0 \\ 0, & \text{if } \lambda_{kt}^i = 0 \text{ and } g_{kt}^i < 0 \end{cases}$.

Table 3 Patients' planned appointment time and expected finished time

Patient id	Workflow id	Planned appointment time	Expected completion time
1	3	2016/10/20-08:15	2016/10/20-09:00
2	3	2016/10/20-09:45	2016/10/20-10:30
3	3	2016/10/20-09:25	2016/10/20-10:10
4	2	2016/10/20-10:25	2016/10/20-11:10
5	2	2016/10/20-09:55	2016/10/20-10:40
6	2	2016/10/20-11:15	2016/10/20-11:40
7	2	2016/10/20-11:30	2016/10/20-11:55
8	3	2016/10/20-11:40	2016/10/20-12:25
9	3	2016/10/20-11:45	2016/10/20-12:30
10	3	2016/10/20-12:00	2016/10/20-12:45

Table 4 Earliness/Lateness Penalty coefficients

Patient id	Earliness cost	Lateness cost
1	20	40
2	20	40
3	20	40
4	20	40
5	20	40
6	20	40
7	20	40
8	20	40
9	20	40
10	20	40

As the number of Lagrange multipliers are proportional to $\mathrm{K_n}$ and T, the computation time involved in updating λ will increase approximately with $\mathrm{K_n}T$. Therefore, the overall computational complexity is $O(\mathrm{K_n^2}T^2 d_p)$. This indicates the computational complexity of our algorithm is polynomial with respect to problem size.

For the centralized approach, we define the centralized patient scheduling problem as follows. If we solve the scheduling problem for based on a centralized computing architecture, the Petri net models of all workflow agents will be merged first. Let Ω denote the set of workflow agents required for a complete medical workflows. The centralized patient scheduling problem $CPSP$ for the complete medical workflows $CW = \|_{n\in\Omega} W_n$ is formulated similarly. $CPSP$ is also defined based on the requirements of patient agents. But the flow balance equations are defined based on CW instead of W_n.

As the number of nodes in the network associated with $L(\lambda)$ is proportional to $\left(\sum_{\mathrm{n}} \mathrm{K_n}\right) T$ and the flow is $D = \sum_{p} d_p$, the computational complexity to compute $L(\lambda)$ is bounded by $O(|\Omega|^2\, \mathrm{K}^2 T^2 D)$. As the number of

Table 5 Patients' arrival time

Patient id	Workflow id	Arrival time
1	3	2016/10/20-8:10
2	3	2016/10/20-9:40
3	3	2016/10/20-9:30
4	2	2016/10/20-10:40
5	2	2016/10/20-10:35
6	2	2016/10/20-11:40
7	2	2016/10/20-11:55
8	3	2016/10/20-11:30
9	3	2016/10/20-11:20
10	3	2016/10/20-11:25

Table 6 Comparison with FCFS

Patient id	Total time (our approach) (min.)	Total time (FCFS) (min.)	Total time (FCFS)/ Total time (our approach)
1	45	45	100%
2	45	65	133.3%
3	45	55	122.2%
4	25	40	160.0%
5	25	25	100%
6	25	70	280%
7	25	75	300%
8	45	85	188%
9	60	45	75%
10	65	65	100%

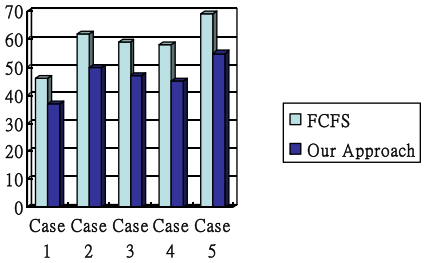


Fig. 11 Comparison with FCFS (*in minute*)

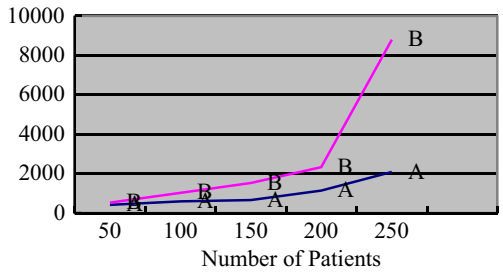


Fig. 12 Response time with respect to the number of patients. (**a**: Our agent based approach, **b**: Centralized approach)

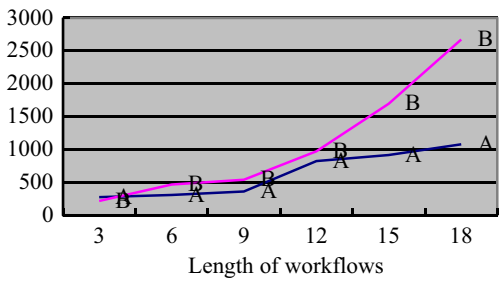


Fig. 13 Response time with respect to the length of workflows

Lagrange multipliers are proportional to $\left(\sum_{n} K_n\right) T$, the computation time involved in updating λ will increase approximately with $O\left(\left(\sum_{n} K_n\right) T\right)$ and is bounded by $O(|\Omega| KT)$. Therefore, the overall response time for centralized architecture will be bounded by $O(|\Omega|^2 K^2 T^2 D_n)$.

Figure 12 shows response time with respect to the number of patients for our agent based approach and a centralized problem solver, CPLEX. Obviously, our approach is more efficient than the centralized problem solver. Figure 13 shows the growth of response time with respect to the length of workflows. As expected, the response time of our agent based approach is much less than that of centralized approach as the length of workflows grows.

8 Conclusion

The healthcare sector faces an intense and growing competitive pressure. How to take advantage of the cutting-edge technologies to effectively acquire competitive advantage is critical for healthcare service providers to survive. Scheduling patients in hospitals is an important issue. Development of a sustainable methodology that combines the state of the art technologies with new organizational structure and management strategy to effectively and dynamically schedule medical operations is required. Recent trends on scheduling operations in hospitals concentrate on the development of dynamic and distributed scheduling techniques to rapidly respond to changing need and requests of patients and achieve sustainability. The problem to optimize the schedule to handle patients using the available resources in a hospital can be formulated as a patient scheduling problem in distributed environment. In this paper, we propose a sustainable solution methodology to dynamically optimize and schedule medical activities based on multi-agent system architecture and collaboration of agents to meet the requirements of patients.

To achieve sustainability, flexibility and scalability, several requirements must be met. First, the medical procedure in a hospital must be described and specified by a standard format. Second, the multi-agent system platform used for implementation must also support information infrastructure and interaction protocols/mechanism defined by industrial standard organization to attain interoperability between agents. Third, the scheduling algorithm must be developed based on dynamic publication and discovery of resources. Fourth, the scheduling method must take advantage of the distributed computing architecture to achieve scalability by solving the scheduling problem based on a divide and conquer strategy. Fifth, all the information in the negotiation processes, including call for proposals, proposals, awarding of contracts and establishment of contracts, must

be described based on a standard format such as XML. Our proposed methodology meets all the above mentioned requirements by (1) using Petri net as the workflow specification language, (2) adopting JADE, a FIPA compliant multi-agent platform that supports ACL and the contract net protocol (CNP), (3) using the publication/discovery infrastructure provided in the JADE platform, (4) developing scalable algorithms based on distributed computing architecture to solve the scheduling problem and (5) specifying all the messages of CNP in XML and designing the associated XML schemas.

We propose a viable and systematic approach to develop a system for scheduling patients in a hospital based on MAS. Workflows and resources such as doctors, staffs, specialists and nurses are all modeled as agents. The key issue is to make these agents work together coherently to handle patients timely. Our approach schedules patients based on dynamically discovered resources. As Petri net is adopted as a workflow specification language in our scheduling system, users do not have to formulate the scheduling problem manually. In this way, users not familiar with optimization theories can also apply our tool to schedule patients in a hospital. Moreover, the cost and time involved in the development of scheduling software can be significantly reduced. In our solution methodology, the scheduling problem is divided into several scheduling subproblems that are solved by several cooperative scheduling agents, with one scheduling subproblem being automatically formulated by a scheduling agent based on the PNML models for the relevant workflow agent and resource agents. Each scheduling subproblem is then solved internally based on the developed algorithm.

We present our implementation and illustrate the functions of the proposed system. We demonstrate the effectiveness and scalability of our method by application scenarios and compare the schedules generated by our method with those generated by the heuristic rule used in practice for scheduling patients. The results indicate that our approach outperforms the heuristic rule. We also analyze and illustrate scalability of our approach by examples. To study scalability of our approach, we analyze response time with respect to the number of patients and the length of workflows. Both our analysis and numerical results indicate that our approach is more efficient than centralized problem solver as the number of patients and the length of workflows grow. One of our future research directions is to develop a scheme to improve overall system performance and relationships between agents by creating different scenarios using different organizational structure of agents in our proposed system. Another future research direction is to study proper mechanisms to form coalitions for agents in a hospital to handle and schedule patients. How to develop an effective scheme to deal with uncertainties in hospitals is also an important issue.

Acknowledgement This paper is currently supported in part by Ministry of Science and Technology, Taiwan under Grant MOST 105-2410-H-324-005.

Appendix I: Representation of workflows, requests and resources

Table 7 XML file, Workflow.xml, for workflow

Workflow.xml

```
<?xml version ="1.0" encoding="UTF-8"
standalone="yes"?>
<Workflow>
  <Workflow_ID>Workflow1</Workflow_ID>
  <WorkflowType>Surgery</WorkflowType>
  <ModelFile>W1.pnml</ModelFile>
  <ProductType>
    <InputType>null,</InputType>
    <OutputType>1</OutputType>
  </ProductType>
  <Requirements>
    <RequiredActivities>
      <Activity>
        <ActivityNumber>1</ActivityNumber>
        <StartTransition>t1</StartTransition>
        <EndTransition>t2</EndTransition>
        <Location>
          <Start>
            <StartPlace>p1</StartPlace>
            <Latitude>null</Latitude>
            <Longitude>null</Longitude>
          </Start>
          <End>
            <EndPlace>p3</EndPlace>
            <Latitude>null</Latitude>
            <Longitude>null</Longitude>
          </End>
        </Location>
      </Activity>
    </RequiredActivities>
  </Requirements>
  <Update_TimeStamp>
    <Year>2014</Year>
    <Month>2</Month>
    <Day>19</Day>
    <Hr>12</Hr>
    <Min>38</Min>
    <Sec>10</Sec>
  </Update_TimeStamp>
</Workflow>
```

Table 8 XML file, PatientRequest.xml, for a patient request

PatientRequest.xml

```
<?xml version="1.0" encoding="UTF-8"
standalone="yes"?>
<OrderRequest>
  <Patient_Request_ID>1</Patient_Request_ID>
  <Order>
    <Order_ID>Patient1</Order_ID>
    <Request>
      <WorkflowType>3</WorkflowType>
      <DueDate>
        <Year>2014</Year>
        <Month>2</Month>
        <Day>20</Day>
        <Hr>9</Hr>
        <Min>0</Min>
      </DueDate>
      <EarlinessPenalty-
  Cost>20</EarlinessPenaltyCost>
      <TardinessPenalty-
  Cost>40</TardinessPenaltyCost>
      <DemandQuantity>1</DemandQuantity>
    </Request>
  </Order>
  <Order_TimeStamp>
    <Year>2014</Year>
    <Month>2</Month>
    <Day>19</Day>
    <Hr>13</Hr>
    <Min>23</Min>
    <Sec>35</Sec>
  </Order_TimeStamp>
</OrderRequest>
```

Table 9 An XML file for describing the capability of a resource agent

Resource.xml

```
<?xml version="1.0" encoding="UTF-8"
standalone="yes"?>
<Resource>
  <Resource_ID>Resource1</Resource_ID>
  <ModelFile>R1.pnml</ModelFile>
  <Location>
    <Latitude>null</Latitude>
    <Longitude>null</Longitude>
  </Location>
  <CanPerformActivities>
    <Activity>
      <ActivityNumber>1</ActivityNumber>
```

Table 9 (continued)

```
      <Transition>
        <Start>
          <StartTransition>t1</StartTransition>
          <StartFiringTime>2</StartFiringTime>
        </Start>
        <End>
          <EndTransition>t2</EndTransition>
          <EndFiringTime>3</EndFiringTime>
        </End>
      </Transition>
      <Capacity>1</Capacity>
    </Activity>
  </CanPerformActivities>
  <Update_TimeStamp>
    <Year>2014</Year>
    <Month>2</Month>
    <Day>19</Day>
    <Hr>12</Hr>
    <Min>33</Min>
    <Sec>40</Sec>
  </Update_TimeStamp>
</Resource>
```

Appendix II: Workflow schedules for patients

Table 10 Workflow Schedule for Patient 1

Workflow	Activity	Start time	End time
1	1	2016/10/20-08:15	2016/10/20-08:20
2	2	2016/10/20-08:20	2016/10/20-08:40
3	3	2016/10/20-08:40	2016/10/20-09:00

Table 11 Workflow Schedule for Patient 2

Workflow	Activity	Start time	End time
1	1	2016/10/20-09:45	2016/10/20-09:50
2	2	2016/10/20-09:50	2016/10/20-10:10
3	3	2016/10/20-10:10	2016/10/20-10:30

Table 12 Workflow Schedule for Patient 3

Workflow	Activity	Start time	End time
1	1	2016/10/20-09:30	2016/10/20-09:35
2	2	2016/10/20-09:35	2016/10/20-09:55
3	3	2016/10/20-09:55	2016/10/20-10:15

Table 13 Workflow Schedule for Patient 4

Workflow	Activity	Start time	End time
1	1	2016/10/20-10:45	2016/10/20-10:50
2	2	2016/10/20-10:50	2016/10/20-11:10

Table 14 Workflow Schedule for Patient 5

Workflow	Activity	Start time	End time
1	1	2016/10/20-10:35	2016/10/20-10:40
2	2	2016/10/20-10:40	2016/10/20-11:00

Table 15 Workflow Schedule for Patient 6

Workflow	Activity	Start time	End time
1	1	2016/10/20-11:15	2016/10/20-11:20
2	2	2016/10/20-11:20	2016/10/20-11:40

Table 16 Workflow Schedule for Patient 7

Workflow	Activity	Start time	End time
1	1	2016/10/20-11:30	2016/10/20-11:35
2	2	2016/10/20-11:35	2016/10/20-11:55

Table 17 Workflow Schedule for Patient 8

Workflow	Activity	Start time	End time
1	1	2016/10/20-11:40	2016/10/20-11:45
2	2	2016/10/20-11:45	2016/10/20-12:05
3	3	2016/10/20-12:05	2016/10/20-12:25

Table 18 (i) Workflow Schedule for Patient 9

Workflow	Activity	Start time	End time
1	1	2016/10/20-11:45	2016/10/20-11:50
2	2	2016/10/20-12:05	2016/10/20-12:25
3	3	2016/10/20-12:25	2016/10/20-12:45

Table 19 Workflow Schedule for Patient 10

Workflow	Activity	Start time	End time
1	1	2016/10/20-12:00	2016/10/20-12:05
2	2	2016/10/20-12:25	2016/10/20-12:45
3	3	2016/10/20-12:45	2016/10/20-13:05

Table 20 Workflow Schedule for Patient 1 (FCFS)

Workflow	Activity	Start time	End time
1	1	2016/10/20-08:15	2016/10/20-08:20
2	2	2016/10/20-08:20	2016/10/20-08:40
3	3	2016/10/20-08:40	2016/10/20-09:00

Table 21 Workflow Schedule for Patient 2 (FCFS)

Workflow	Activity	Start time	End time
1	1	2016/10/20-09:40	2016/10/20-09:45
2	2	2016/10/20-10:05	2016/10/20-10:25
3	3	2016/10/20-10:25	2016/10/20-10:45

Table 22 Workflow Schedule for Patient 3 (FCFS)

Workflow	Activity	Start time	End time
1	1	2016/10/20-09:30	2016/10/20-09:35
2	2	2016/10/20-09:45	2016/10/20-10:05
3	3	2016/10/20-10:05	2016/10/20-10:25

Table 23 Workflow Schedule for Patient 4 (FCFS)

Workflow	Activity	Start time	End time
1	1	2016/10/20-10:40	2016/10/20-10:45
2	2	2016/10/20-11:00	2016/10/20-11:20

Table 24 Workflow Schedule for Patient 5 (FCFS)

Workflow	Activity	Start time	End time
1	1	2016/10/20-10:35	2016/10/20-10:40
2	2	2016/10/20-10:40	2016/10/20-11:00

Table 25 Workflow Schedule for Patient 6 (FCFS)

Workflow	Activity	Start time	End time
1	1	2016/10/20-11:40	2016/10/20-11:45
2	2	2016/10/20-12:30	2016/10/20-12:50

Table 26 Workflow Schedule for Patient 7 (FCFS)

Workflow	Activity	Start time	End time
1	1	2016/10/20-11:55	2016/10/20-12:00
2	2	2016/10/20-12:50	2016/10/20-13:10

Table 27 Workflow Schedule for Patient 8 (FCFS)

Workflow	Activity	Start time	End time
1	1	2016/10/20-11:30	2016/10/20-11:35
2	2	2016/10/20-12:05	2016/10/20-12:30
3	3	2016/10/20-12:30	2016/10/20-12:55

Table 28 Workflow Schedule for Patient 9 (FCFS)

Workflow	Activity	Start time	End time
1	1	2016/10/20-11:20	2016/10/20-11:25
2	2	2016/10/20-11:25	2016/10/20-11:45
3	3	2016/10/20-11:45	2016/10/20-12:05

Table 29 Workflow Schedule for Patient 10 (FCFS)

Workflow	Activity	Start time	End time
1	1	2016/10/20-11:25	2016/10/20-11:30
2	2	2016/10/20-11:45	2016/10/20-12:05
3	3	2016/10/20-12:05	2016/10/20-12:30

References

1. Decker K, Jinjiang L (1998) Coordinated hospital patient scheduling. In: International conference on multi agent systems, pp 104–111, Paris
2. Kutanoglu E, Wu SD (1999) On combinatorial auction and Lagrangean relaxation for distributed resource scheduling. IIE Trans 31:813–826
3. Oddi A, Cesta A (2000) Toward interactive scheduling systems for managing medical resources. Artif Intell Med 20:113–138
4. Daknou A, Zgaya H, Hammadi S, Hubert H (2010) A dynamic patient scheduling at the emergency department in hospitals. In: 2010 IEEE workshop on health care management, pp 1–6
5. Spyropoulos CD (2000) AI Planning and scheduling in the medical hospital environment. Artif Intell Med 20:101–111
6. Nilsson NJ (1998) Artificial intelligence: a new synthesis. Morgan Kaufmann Publishers Inc., San Francisco
7. Ferber J (1999) Multi-Agent Systems an introduction to distributed artificial Intelligence. Addison-Wesley, Reading, MA
8. Durfee EH, Lesser VR, Corkill DD (1989) Trends in cooperative distributed problem solving. IEEE Trans Knowl Data Eng 1(1):63–83
9. Russel SJ, Norvig P (2006) Artificial intelligence—a modern approach, 2nd ed., Pearson Education Asia Limited
10. Workflow Management Coalition (2009) XPDL support and resources. http://www.wfmc.org/xpdl.html
11. Object Management Group (2009) Business process modeling notation. http://www.bpmn.org
12. OASIS (2009) Web services business process execution language version 2.0. http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html
13. Merlin P, Farbor D (1976) Recoverability of communication protocols. IEEE Trans Commun 24(9):1036–1043
14. Smith RG (1980) The Contract net protocol: high-level communication and control in a distributed problem solver. IEEE Trans Comput 29:1104–1113
15. Murata T (1989) Petri nets: properties, Analysis and Applications. Proc IEEE 77(4):541–580
16. Güray Güler M (2013) A hierarchical goal programming model for scheduling the outpatient clinics. Expert Syst Appl 40(12):4906–4914
17. Conry SE, Kuwabara K, Lesser VR, Meyer RA (1991) Multistage negotiation for distributed constraint satisfaction. IEEE Trans Syst Man Cybern 21(6):1462–1477
18. Marinagia CC, Spyropoulosa CD, Papatheodoroub C, Kokkotos S (2000) Continual planning and scheduling for managing patient tests in hospital laboratories. Artif Intell Med 20:139–154
19. Edmonds J, Karp RM (1972) Theoretical improvements in algorithmic efficiency for network flow problems. J ACM 19(8):248–264
20. McFarlane DC, Bussmann S (2000) Developments in holonic production planning and control. Int J Prod Plan Control 11(6):522–536
21. Parunak HVD (1987) Manufacturing experiences with the contract net. In: Huhns M (ed) distributed artificial intelligence. London, Pitman, pp 285–310
22. Ramos C (1996) A holonic approach for task scheduling in manufacturing systems. In: Proceedings of the 1996 IEEE international conference on robotics and automation, pp 2511–2516
23. Brennan RW, Norrie DH (2001) Evaluating the performance of reactive control architectures for manufacturing production control. Comput Ind 46(9):235–245
24. Neligwa T, Fletcher M (2003) An HMS operational model. In: Deen SM (ed) Agent-based manufacturing: advances in the holonic approach. Springer, Berlin, pp 163–191
25. Sandholm TW (1998) Contract types for satisfying task allocation: I theoretical results. In: AAAI Spring symposium series: satisficing models. Stanford University, CA, pp 68–75
26. van der Aalst WMP (1998) The application of Petri nets to workflow management. J Circuit Syst Comput 8(1):21–66
27. van der Aalst WMP, Kumar A (2001) A reference model for team-enabled workflow management systems. Data Knowl Eng 38(9):3355–363
28. Weske M, van der Aalst WMP, Verbeek HMW (1998) Advances in business process management. Data Knowl Eng 50(1):1–8
29. Weber M, Ekkart K (2012) The Petri Net Markup Language. http://www2.informatik.hu-berlin.de/top/pnml/download/about/PNMLLNCS.pdf
30. Billington J, Christensen S, van Hee K, Kindler E, Kummer O, Petrucci L, Post R, Stehno C, Weber M (2003) The petri net markup language: concepts, Technology, and Tools. Lect Notes Comput Sci 2679:483–505
31. Hsieh FS, Lin JB (2014) Development of context-aware workflow systems based on Petri Net Markup Language. Comput Stand Interfaces 36(9):672–685
32. Polyak BT (1969) Minimization of unsmooth functionals, USSR computational. Math Math Phys 9:14–29
33. Hsieh FS, Lin JB (2014) Scheduling patients in hospitals based on multi-agent systems In: Moonis A et al. (eds) IEA/AIE 2014, Part I, LNAI 8481, pp 32–42
34. Saremi A, Jula P, ElMekkawy T, Wang GG (2013) Appointment scheduling of outpatient surgical services in a multistage operating room department. Int J Prod Econ 141(8):646–658
35. Sauré A, Patrick J, Tyldesley S, Puterman ML (2012) Dynamic multi-appointment patient scheduling for radiation therapy. Eur J Oper Res 223(8):573–584

36. Ceschia S, Schaerf A (2012) Modeling and solving the dynamic patient admission scheduling problem under uncertainty. Artif Intell Med 56(9):199–205
37. Boudoulas KD, Leier CV, Geleris P, Boudoulas H (2015) The shortcomings of clinical practice guidelines. Cardiology (Switzerland) 130:187–200
38. Cecamore C, Savino A, Salvatore R, Cafarotti A, Pelliccia P, Mohn A, Chiarelli F (2011) Clinical practice guidelines: What they are, why we need them and how they should be developed through rigorous evaluation. Eur J Pediatr 170:831–836
39. Fox J, Patkar V, Chronakis I, Begent R (2009) From practice guidelines to clinical decision support: closing the loop. J Royal Soc Med 102:464–473
40. Kaiser K, Marcos M (2016) Leveraging workflow control patterns in the domain of clinical practice guidelines. BMC Med Inform Decis Mak 16:20
41. ten Teije A, Miksch S, Lucas P (2008) Computer-based clinical guidelines and protocols: a primer and current trends. IOS Press, Amsterdam
42. Isern D, Moreno A (2016) A systematic literature review of agents applied in healthcare. J Med Syst 40(40):43
43. Kirn S, Herrler R, Heine C, Krempels KH (2003) Agent.Hospital - agent-based open framework for clinical applications. In: Proceedings of the 12th IEEE international workshops on enabling technologies: infrastructure for collaborative enterprises, pp 36–41
44. Kirn S, Anhalt C, Krcmar H, Schweiger A (2006) Agent.Hospital — health care applications of intelligent agents. In: Kirn S, Herzog O, Lockemann P, Spaniol O (eds) multiagent engineering: theory and applications in enterprises, international handbook on information systems. Springer, Berlin Heidelberg, pp 199–220
45. Hutzschenreuter AK, Bosman PAN, Blonk-Altena I, van Aarle J, La Poutre JA (2008) Agent-based patient admission scheduling in hospitals. In: Proceedings of 7th international joint conference on autonomous agents and multiagent systems: industrial track. Estoril, Portugal, pp 45–52
46. Braubach L, Pokahr A, Lamersdorf W (2014) Negotiation-based patient scheduling in hospitals - reengineering message-based interactions with services. In: Iantovics B, Kountchev R (eds) Advanced intelligent computational technologies and decision support systems, studies in computational intelligence. Springer Verlag, vol 486, pp 107–121
47. (2014) Object Management Group Model Driven Architecture (MDA) MDA Guide rev. 2.0, http://www.omg.org/cgi-bin/doc?ormsc/14-06-01
48. Selic B (2013) The pragmatics of model-driven development. IEEE Softw 20(5):19–25

Fu-Shiung Hsieh received the B.S. and M.S. degrees in control engineering from National Chiao-Tung University, Taiwan, Republic of China in 1987 and 1989, respectively. He received the Ph.D. Degree from National Taiwan University in 1994. He served as research fellow in Industrial Technology Research Institute (ITRI), Taiwan, from 1994 to 1999. He was an Assistant Professor and Associate Professor with The Overseas Chinese Institute of Technology from 1999 to July 2004 and August 2004 to July 2005, respectively. Since August 2005, he has been with Chaoyang University of Technology, where he is currently a Professor at the Department of Computer Science and Information Engineering. He has over eighty publications, including international journal and conference papers. He served as a Guest Editor for Asian Journal of Control. He has also served as a referee for Automatica, Fuzzy sets and systems, IEEE Transactions on Systems, Man, & Cybernetics, IEEE Transactions on Robotics & Automation and Journal of Information Science and Engineering, Journal of Intelligent Manufacturing, International Journal of Production Research, International Journal of Advanced Manufacturing Technology, Measurement, Asian Journal of Control, International Journal of Control, Automation in Construction, and IEEE Transactions on Industrial Informatics. He chaired/co-chaired several international conferences, including 2001 and 2006 IEEE International Conference on Systems, Man& Cybernetics, IFAC 2008 World Congress, 2009 International Conference on Flexible Automation and Intelligent Manufacturing, 2010 International Conference on Information Society (i-Society 2010), International Conference on Computational Collective Intelligence — Technology and Applications (ICCCI 2010), 2012 International Conference on New Trends in Information Science, Service Science and Data Mining (ISSDM 2012), the 10th International Conference on Service Systems and Service Management (ICSSSM 13) and the 27th International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA-AIE 2014). He is listed in Marquis Who's Who in the World 2006, 2007 and 2009 and Marquis Who's Who in Asia 2007 and 2012. His area of interest includes electronic commerce, workflows, multi-agent systems, intelligent systems and manufacturing systems.