SOCA (2007) 1:185–196
DOI 10.1007/s11761-007-0012-1

ORIGINAL PAPER

Multi-agent Pareto appointment exchanging in hospital patient scheduling

Ivan Vermeulen · Sander Bohte · Koye Somefun · Han La Poutré

Received: 6 May 2007 / Accepted: 13 July 2007 / Published online: 16 August 2007

Abstract We present a dynamic and distributed approach to the hospital patient scheduling problem, in which patients can have multiple appointments that have to be scheduled to different resources. To efficiently solve this problem we develop a multi-agent Pareto-improvement appointment exchanging algorithm: MPAEX. It respects the decentralization of scheduling authorities and continuously improves patient schedules in response to the dynamic environment. We present models of the hospital patient scheduling problem in terms of the health care cycle where a doctor repeatedly orders sets of activities to diagnose and/or treat a patient. We introduce the Theil index to the health care domain to characterize different hospital patient scheduling problems in terms of the degree of relative workload inequality between required resources. In experiments that simulate a broad range of hospital patient scheduling problems, we extensively compare the performance of MPAEX to a set of scheduling benchmarks. The distributed and dynamic MPAEX performs almost as good as the best centralized and static scheduling heuristic, and is robust for variations in the model settings.

Keywords Health care · Patient scheduling · Multi-agent systems

1 Introduction

The first thing that one encounters when seeking medical assistance in a hospital is a schedule: the scheduled medical professionals to consult, time-slots for possible diagnostic or therapeutic machines, and availability of simple resources like examination rooms. Depending on the available capacity, these schedules may be more or less congested. In particular, in countries like The Netherlands and Greece, demand regularly exceeds capacity and substantial waiting lists exist for many medical procedures. However, even in these situations, it is not the case that resources are always used at high efficiency. Medical professionals report many schedule inefficiencies. Effective scheduling algorithms should decrease waiting-lists significantly, while increasing hospital efficiency [2,3].

Traditional industrial scheduling techniques are studied in the field of operations research (OR). OR techniques are very effective for solving well-defined centralized optimization problems, where the algorithm can determine the optimal schedule for all parties involved. Scheduling solutions based on OR have been implemented in specific targeted health-care problems, like staff planning (for an overview, see [4]). However, OR techniques have so far found little favor in hospital patient scheduling, in great part due to the inherent distribution of authority in hospitals [5]. We will discuss the relation of our work with existing OR techniques in more detail in Sect. 6.

In hospital patient scheduling, the scheduling problem is dynamic and in flux: operations take more or less time than anticipated, crucial staff may not be available, equipment breaks down, and new urgent patients arrive. Not unimportantly, hospitals are also organized around different autonomous departments (wards, ancillary units) each with their own specialty, and each department essentially has authority

A preliminary version of this work has appeared as [1].

I. Vermeulen (✉) · S. Bohte · K. Somefun · H. La Poutré
CWI, Centre for Mathematics and Computer Science,
Kruislaan 413, 1098 SJ, Amsterdam, The Netherlands
e-mail: vermeule@cwi.nl

S. Bohte
e-mail: sbohte@cwi.nl

K. Somefun
e-mail: koye@cwi.nl

H. La Poutré
e-mail: hlp@cwi.nl

over their own schedule. Thus, hospital scheduling has strong decentralized characteristics.

The dynamic nature of hospital patient scheduling, together with the decentralization of scheduling authority, suggests that a more suitable approach to hospital patient scheduling may be one that fits the problem domain better: a distributed multi-agent system [6]. Here, we take a first step toward developing a multi-agent hospital agent scheduling solution that respects the current distribution of scheduling authority, and that is capable of continuously adjusting the different schedules in response to the dynamic environment.

The general idea of an agent system [7] is that each party—e.g., doctors, patients, resources—is represented by a software entity—the agent—that acts autonomously on behalf of its owner. Each agent "knows" the preferences and constraints of its owner. The objective of a multi-agent scheduling system is to design the agents and the interaction rules such that together, the agents can arrive at effective schedules.

Related work has studied parts of this multi-agent scheduling problem: Decker and Li [8] consider the problem of resource conflicts in patient scheduling. They design a specific interaction mechanism for resource-representing software agents to prevent such conflicts. The implementation of such a mechanism is shown to achieve substantial productivity gains.

Some distributed constraint solving approaches are also specifically applied to health care scheduling. In [9], resource and patients constraints and preferences (soft constraints) are formally represented and agents use off-the-shelf constraints solvers locally, but a central solver to coordinate constraints is also needed. Additionally, this is a static approach as it only solves an initial scheduling problem and then appointments are assumed to be fixed after that.

Given the distributed and decentralized nature of hospital patient scheduling, the use of a market mechanism for scheduling seems a more natural fit: markets can efficiently distribute scarce resources, they can facilitate dynamic environments, and only price-quotes need to be exchanged between participants, rather than complex constraints and preferences. In [10] a first step is taken in developing a framework for using virtual markets to solve distributed scheduling problems. Different types of auction mechanisms (and additionally bidding strategies [11]) are analyzed. The results show that it is hard to find a general solution. Alternatively in [12] a contracting model for agent-based scheduling is discussed. The search space in this case is very large and a search bias must be set accurately.

In [13] a multi-agent approach is suggested to solve scheduling in hospitals. The authors distinguish multiple planning steps, and consider negotiation between agents for a schedule improvement. It provides theoretical insight, but does not discuss any implementation. A further developed market-based approach to hospital patient scheduling is taken by Paulussen et al. [5]. They introduce software agents that represent the *interests* of the patients, and resources where medical actions take place are represented by resource agents. To distribute the resources amongst these patient agents, Paulussen et al. use a market mechanism where patient agents communicate their (private) utility for certain time-slots on a resource via a price mechanism.

Apart from scalability concerns of the system in [5], an additional problem is the use of a utility-function for patient well being. Clearly, when optimizing a schedule the question is what metric to optimize against in a health-care setting, and patient well being is an obvious choice [5]. Quantifying *relative* patient well being, however, is notoriously hard and any choice will be controversial, with both doctors and patients.

Here, we develop a scheduling method that offers substantial gains without having to consider this difficult issue. We observe that often patients have multiple appointments (e.g. [14]). For instance, a patient may need to get a CT scan and an endoscopy, and then consult with a doctor to discuss the results. We note that the time of the *last* appointment effectively determines the patient's "waiting-time" and a scheduling algorithm can potentially move his/her *other* appointments without any negative effect on the waiting time for the patient.

Guaranteeing "not-worse" for schedule changes means that patients actually have an incentive to cooperate. In any practical implementation of a (re)scheduler, patient cooperation will be essential to make sure that patients are actually willing and able to come to the hospital at the new appointment time. Being able to guarantee "not-worse" schedule changes thus gives us an opportunity to improve patient waiting times while avoiding the difficult comparison of which patient's well-being benefits how much from an improvement in waiting time.

Here, we present a multi-agent scheduling method that exploits this opportunity by designing agents that exchange appointment times such that no patient is worse off than before. In economics such a "nobody-worse" improvement is called a Pareto improvement [15]. Thus, we present a Multi-agent Pareto Appointment EXchanging algorithm (MPAEX).

Agents have been proved to be an effective approach to resource allocation, see e.g. [16]. In [17] the authors discuss how socially optimal allocation of resources can be reached. For the scenario similar to MPAEX, where only cooperatively rational deals without side payments are allowed, the authors proof that a sequence of Pareto optimal deals will lead to the Pareto optimal outcome. However, deals of any structural complexity may be necessary to guarantee an optimal outcome.

In the multi-agent system that we develop, patients are assigned an initial schedule for their required treatments. Then, agents acting on behalf of individual patients attempt to exchange the time-slots of the initial appointments with better appointments occupied by other patients. The other patient's agent accepts a proposed exchange of appointments if the resulting schedule is not worse for the patient.

In simulations of (many) hospital patient scheduling problems, we show that when we let patient agents try to improve their patient's schedule, the agents collectively improve the overall patient waiting time.

Based on practical cases, we introduce our hospital patient scheduling model. It represents autonomous departments and resources, as well as individual patients and their activities. We present a semi-dynamic hospital patient scheduling setting to gain fundamental insights and allow comparison with more standard centralized static techniques.

A robust simulation of the distribution of workloads over the various resources in a hospital is a crucial aspect in evaluating scheduling solutions for hospital patient scheduling. For example, the patient scheduling problem may be fundamentally different if either an MRI scan would be a very busy resource, relative to the other resources, or all resources are equally busy. Different hospitals will have different workloads, depending on patient population and the available mix of resources and doctors/staff. To assess accurately how useful different scheduling solutions are, we have to be able to consider a large distribution of different workloads.

As a contribution of this paper, we present a measure for characterizing different workloads: we introduce the Theil index [18] within the hospital scheduling setting. In economics, the Theil index is a common measure of inequality motivated by the notion of entropy. The inequality in a workload can be interpreted as the degree to which bottlenecks are present in the available resources. By conducting experiments for a large range of Theil indices, we obtain a representative sample of the problem space. The characterization of different workloads by their Theil index allows us to compare MPAEX methodically with other (more centralized) scheduling methods.

For these hospital patient-scheduling simulations, we find that for initial schedules generated by a pure first-come-first-served scheduler, exchanging appointment times results in significant improvements for the final schedule after comprehensive appointment exchanging. For initial schedules with more random initial scheduling (first-come-randomly-served), we obtain even better solutions for the final schedule. This random initial scheduling is introduced to better capture the fact that in current practice, the initial schedule is created in part based on (relatively stochastic) patient and resource availability.

We extensively compare the performance of the decentralized MPAEX approach to a set of centralized heuristics and find that MPAEX performs close to centralized scheduling heuristics. Contrary to these centralized techniques, our approach respects the distributive nature of the scheduling authority, takes patient preferences into account, and is well suited for the dynamic nature of hospital scheduling. Additionally, unlike centralized OR approaches and centralized heuristics, MPAEX is suitable for dynamic environments where patients leave and arrive, and where resources are available or off-line.

Finally, the approach explored with MPAEX also provides us with a robust basis: for future research, we want to consider more complex appointment rescheduling environments, the possible inclusion of (artificial) money to enhance exchange possibilities, and more dynamic scheduling settings.

2 Problem

2.1 Hospital patient scheduling

Hospitals are increasingly working with databases that record all scheduled activities. The schedules are typically planned within rosters that are predefined by departments or individual doctors, e.g., a doctor sets specific hours for consulting patients, and consults with patients can only be scheduled in these hours. Authorizations regarding who can access and/or modify the schedules (and rosters) are distributed across the organization depending on local preferences and culture. Most departments are reluctant to allow other departments to make appointments in empty spots in their schedule.

Not surprisingly, many of such electronic scheduling systems are just that: electronic versions of appointment notebooks. Intelligent (re)planning is mostly done by departments, often by hand, and schedule optimization across departments is very hard, as each change needs coordination, usually by phone. Traditional OR techniques are not equipped to deal with coordination between departments with local information, authority and preferences, and fail in such situations. Agent systems are designed for coordination between autonomous parts with local information.

This current practice of decentralized schedule authorizations in particular leads to complications when a number of appointments across different departments need to be scheduled for a patient. It may very well be that simply switching a scheduled patient in one department to a different time may free up the combination of resources that is needed for another patient. The "switched patient" may not even have a preference for one time or another, but this "free" optimization is currently very hard to achieve. In practice, patients that need resources across departments are not effectively scheduled.

2.2 The health care cycle

To study the hospital patient scheduling problem in detail, we consider the current (typical) health care routine.

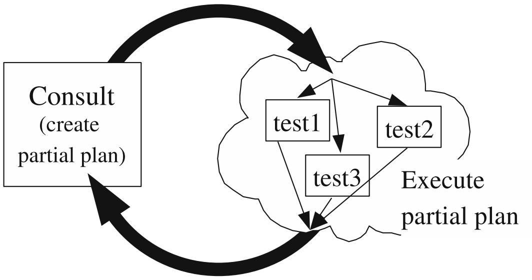

Fig. 1 Consult-diagnostic(s)-consult cycle

Any patient walking into a doctor's office becomes part of the "health care" cycle: if a medical problem is suspected, a number of actions will be scheduled to diagnose the exact nature of the problem and/or a treatment plan is scheduled. Central in the health care cycle is the doctor treating the patient: the doctor first requests diagnostics or treatments, and upon completion, the results and patient return to the doctor. In the consult, the doctor then decides what activities must take place next.

We call the set of activities ordered a *partial plan*, see Fig. 1. A partial plan can consists of a number of different activities, possibly involving different resources and with constraints between them. Scheduling patient activities is a complex task: diagnostic tests may require the cooperation of a number of people and resources in the hospital, so appointments have to be scheduled at times when all these resources are available (and the patient of course).

2.3 Patient treatment plans

The scheduling complexities for typical patient treatment plans that need resources across departments can be illustrated by the workflow in two cases that are typically hard to schedule efficiently. These cases are taken from the protocols of the Oncology department of the Academic Medical Center in Amsterdam. The consult-diagnostics-consult cycle and the partial plans comprising of multiple diagnostics are clearly discernable in both cases.

Diagnosis gallbladder obstruction In Fig. 2, the treatment plan for a patient with an obstructed gallbladder pathway is shown (taken from the protocols for gallbladder obstruction at the Academic Medical Center, Amsterdam). It consists of a series of successive partial plans where at each point the doctor decides what to do next, based on the diagnostic results. A patient with gallbladder obstruction is first diagnosed with an echo + Doppler for the veins to diagnose for gallstones. If this is the case, treatment can start. Otherwise, to consider the possibility of a tumor, one of three diagnostic plans is selected: either an ERCP + Stent, a PTC/PTD test, or a set of four diagnostics (an endo-echo, a CT-scan, a MRI, and an MRCP) is ordered. From this point again, the patient goes through progressive doctor-diagnosis cycles to determine how to operate.

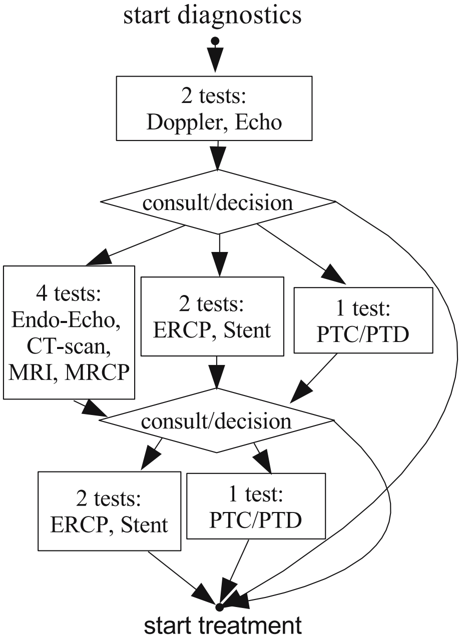

Fig. 2 Diagnostic strategy for gallbladder obstruction

Diagnosis breast cancer Any persisting breast abnormality requires further diagnostics: the best evaluation of (palpable) mammolesions is through three independent examinations by a surgeon, a radiologist, and a (cyto)pathologist (taken from the protocols for breast cancer at the Academic Medical Center, Amsterdam). Each specialist gives an independent assessment that are then equally weighted. If this triple diagnostic suspects a malignant tumor, further (definitive) diagnostics are scheduled, such as a histological or cytological needle biopsy. Upon a definitive positive diagnosis, it is determined how to operate, where further diagnostic tests may be considered depending on the severity of the tumor.

2.4 Model

We abstract hospitals to a set of resources (including staff), constraints, staff preferences, budget considerations. A hospital is divided into different departments. These departments have a level of autonomy in their actions, and between them, they have common as well as self-interested goals. Departments want to work as efficiently as possible, and they want to take their staff's preferences into account. We model resources on the level of the combined requirements for a single schedule, e.g., an entire warden, or the MRI-schedule including the associated staff etc. From the medical cases described in the previous subsection, we can distill a number of stylized facts.

Patient partial plan With respect to the patient plan, we model a consult as consisting of a limited number of subtasks or activities. These activities can usually be performed in any order, in our model there are no precedence constraints (as in the cases described in Sect. 2.3). Once all activities have been performed, the responsible doctor determines whether additional activities are necessary. For our approach, it is irrelevant whether activities are new or additional; therefore, we will not make the distinction. Henceforth we will call a number of activities issued at the same time for a single patient by a doctor a partial plan. Based on the result of this partial plan the doctor may issue more activities in the future. We label this new set of activities as a new partial plan. Since most activities within a partial plan are performed on a patient, we assume not more than one activity can be performed at the same time on one patient, hence the different activities in the partial plan have to be scheduled for different times.

Duration of activities Different activities make use of different resources. Consequently, the standard time, which is reserved for such an activity, may differ from one resource to the other. For example, the time necessary for taking an X-ray or performing an ultrasound examination may differ. We assume that all activities on a resource require the same type of appointment, and therefore take the same amount of time. Between resources, the activity time varies.

Objective The hospital wants to minimize the completion times of all patients, given the restriction that within a treatment activities cannot be performed at the same time. To achieve this objective, online decisions about scheduling and especially *rescheduling* are needed that improve the throughput of patients. Completion time of a patient is the time from the creation of the partial plan to completion time of the final activity from this plan.

3 Multi-agent pareto appointment exchanging

To schedule the patient activities efficiently, we use a distributed approach where software agents interact with each other to exchange appointments where none of two interacting parties is worse off: the agents are *Pareto*-improvers. Worse-off is defined in subjective terms, as each agent acts according to its individual constraints and tries to optimize preferences. Our scheduling method thus amounts to MPAEX. We have two types of agents: resource agents and patient agents.

Each resource agent represents one resource, and it takes into account constraints like fixed hours, and preferences like preferring not to require over-time. When scheduling an activity to a timeslot on a resource, the resource agent makes sure no constraints are violated and tries to optimize preferences over resource schedules.

Algorithm 1 Initial timeslot assignment.

```
For every patient agent:
1: while not all activities are scheduled do
2:   select an unscheduled activity from partial plan
3:   ask corresponding resource agent for a timeslot
4:   if the timeslot is not conflicting then
5:     accept timeslot
6:   else
7:     goto 3
```

Algorithm 2 Multi-Agent Pareto Appointment EXchanging (MPAEX)

```
For every patient agent: (until no more exchanges can be made)
 1: select an activity to reschedule
 2: ask the corresponding resource agent for a feasible prospective time-
    slot
 3: if there are no alternative prospective timeslots then
 4:   goto 1
 5: else if the timeslot is empty then
 6:   accept and reschedule
 7:   goto 1
 8: else
 9:   get identity of patient agent occupying timeslot
10:   propose the exchange to patient agent occupying timeslot
11:   if the resulting schedule of patient agent occupying timeslot is
      not worse than its current schedule then
12:     both agents reschedule
13:     inform resource agent of exchange
14:     goto 1
15:   else
16:     goto 2
```

Each patient is also represented by a software agent—the patient agent—that has knowledge of the patient's needs (activities that need to be scheduled, availability) and preferences (when, which doctor, own schedule). We envision that the medical priorities and partial plans in this agent are determined by the consulting doctor, whereas the patient's preferences, like his/her schedule, are set by the patient. Patient agents make sure none of their activities overlap and try to get the best possible schedule given the patient's individual constraints and preferences. To get an initial schedule for their patient, patient agents interact with resource agents to get timeslots for the activities in their partial plan. To improve their schedules, patient agents interact with each other to exchange time-slots. Resource agents do not interact with each other.

In general there are two processes of scheduling: initial timeslot assignment (Algorithm 1), and patient agents improving their schedules by MPAEX (Algorithm 2). In real-life dynamic settings, these processes run together. Patients arrive one by one and are initially scheduled over the day. Patient agents individually try to improve their patient's schedule, continuously, or at certain events (such as cancellations, lifted resources constraints, passing deadlines).

Algorithm 1 describes patient agents asking the different resource agents one by one an initial timeslot for the activities of their partial plan. The timeslots cannot be conflicting.

Algorithm 2 is implemented by MPAEX: patient agents try to improve their schedule by exchanging appointments for activities. First, the resource agent is contacted to get a feasible prospective timeslot for the selected activity. If this timeslot would improve the schedule but is occupied by another agent, both agents have to agree to exchange the timeslots. No agent will agree to an exchange that will worsen its schedule.

With the approach thus outlined, the scheduling problem can be solved completely distributed. Initial scheduling, as well as MPAEX rescheduling, is done purely by interaction between individually rational agents. In the next section, we discuss how we simulate the patient scheduling problem, and how we generate problem instances.

4 Simulating hospital patient scheduling

4.1 Online hospital patient scheduling

Here, we focus on a semi-dynamic model of hospital patient scheduling instead of a fully dynamic model. The most important dynamic aspect of our semi-dynamic model is that partial plans are scheduled in order of arrival, one partial plan at a time, without knowledge of what partial plans are to be scheduled next (see [19] for this dynamic property of online scheduling). Once all partial plans for all patients are initially scheduled, patient agents will then try to improve their schedule by MPAEX.

Studying this semi-dynamical model first gives us fundamental insights in the performance of our distributed approach relative to alternative approaches. Notably, many state-of-the-art scheduling heuristics are not suitable for fully dynamic and/or distributed scheduling where patients are continuously scheduled and rescheduled. Furthermore, creating a fully dynamic model requires many ad hoc decisions, making the models representative only for very specific situations from which it is not straightforward to generalize to other dynamic scheduling instances.

In practice, the separation between initial scheduling and rescheduling can be more gradual. Our semi-dynamic setting can be thought of as a single iteration of a dynamic system where rescheduling is done periodically after a number of patients are initially scheduled.

Studying this stylized hospital patient scheduling model gives us fundamental insights in the performance of our distributed approach relative to alternative approaches. Although an MPAEX approach can be run continuously as patients are arriving, many state-of-the-art scheduling heuristics are not suitable for such fully on-line and/or distributed scheduling.

To analyze the different scheduling approaches in a general way, and to compare with centralized and static benchmarks, constraints and preferences in our simulation model are as follows: activities can not overlap on a resource schedule, activities of a partial plan must all be scheduled and can not overlap, patient preferences are such that schedules with earlier completion time (C) are preferred.

We use two fundamental methods for initial timeslot assignment (Algorithm 1): First-come-first-served (FCFS), and first-come-randomly-served (FCRS). In FCFS, patients arrive over time and their planned activities are scheduled on the first available time-slot on each required resource. In FCRS, patients arrive over time and their planned activities are scheduled to a random time-slot within a fixed time-window on the required resources.[1] Whereas at first sight FCFS seems to relate most closely to current hospital practice, it ignores the fact that many treatments have limited medical urgency, and the exact date for the appointment is determined both by the patient's schedule and the first availability of the required resources. To reflect this stochastic element of patient and resource availability, we introduce the FCRS schedule. Note that changes in resource schedules may open up new, better appointment opportunities for patients that fit in their schedule (Algorithm 2), so attempting to reschedule still makes sense. Current practice will fall somewhere between FCFS and FCRS, depending on the actual hospital situation.

Given the patient preferences for the schedule of a partial plan (finish as early as possible) we implement Algorithm 2 in MPAEX as follows: Patient agents always select their last activity for rescheduling. Resources agents will then propose an alternative time-slot, starting from the earliest possible timeslot, and the patient agent will try to exchange his/her time-slot with the patient–agent occupying that time-slot. The deal will be accepted if neither patient is worse off according to their preferences, which here means that completion time will not increase. If not accepted, the patient agent will request another prospective time-slot from the resource agent, and will continue doing so until there are no more prospective time-slots, or a proposed exchange is accepted. The process is repeated for all patient agents iteratively until no exchanges can be made any more.

[1] To determine duration of the time-window, FCRS requires a prediction of the number of activities that can be expected on resources so that after randomly handing out time-slots, the capacity available on the resource will be efficiently used; these data are usually available in hospitals.

Table 1 Four schemes of standard activity times for $m = 8$

Scheme name	On 8 resources
Equal	1,1,1,1,1,1,1,1
Small difference	1,1,2,2,3,3,4,4
Reasonable difference	1,2,3,4,5,6,7,8
Large difference	1,3,5,7,9,11,13,15

We found that experiments where an exchange can have more than one activity at a time show very little gain in performance. It is also possible to consider *multilateral* multiple-activities exchanges. The complexity of these types of exchanges quickly becomes intractable. Because of the small complexity, and good performance, of bilateral one-activity appointment exchanges, we only show these results.

4.2 Modeling hospital resource usage

To evaluate the performance of MPAEX fairly, we evaluate hospital patient scheduling performance for a broad distribution of possible hospital characteristics (e.g. [2]).

An important aspect of our model is the workload of the resources relative to each other. In practice, not all resources in a hospital are as busy as others, usually there are a number of crowded resources. The number of patients on the resource, and the time needed for each patient determine the workload of a resource. Different relative workloads influence the problem characteristics, and the performances of scheduling algorithms. Next, we discuss a means to methodically vary these parameters such that we can evaluate a representative cross-section of these properties.

Partial plan We define the distribution over all possible partial plans by a set of the probabilities (P_j) for all resources (j). Let for resource j the probability for a patient to have an activity on this resource, as part of his partial plan, be denoted by P_j. This captures the fact that partial plans can consist of a various number of activities, and that some activities are planned more often than others. The sum over all P_j determines the expected average number of activities per patient. For n patients $n * P_j$ is the expected number of patients on resource j.

Standard activity time We model all activity times on the same resource as equal (standard), however, activity times between resources can vary. Given m resources, we design four schemes of various standard activity times. The schemes range from equal activity times on all resources, to large differences: see Table 1.

In the "equal" scheme all activities on all resources take the same time (unit length). In the other schemes, activities on some resources take (much) more time then activities on other resources. We carry out experiments with these four different schemes.

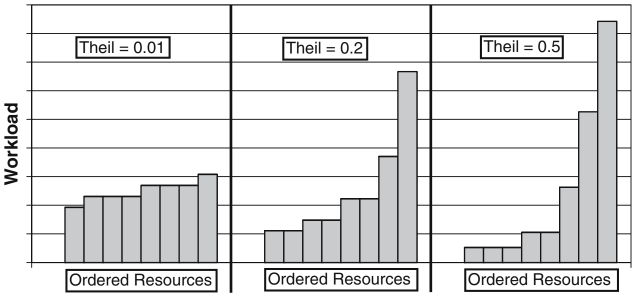

Fig. 3 Examples of Theil index values for three different workload distributions, with $m = 8$. Each *bar* represents the workload on a resource, which are here presented ordered from low to high

Unequal workload distribution The workload of a resource is determined by the number of patients on the resource, and the time needed for each patient (see previous two paragraphs). In our experiments we vary the probabilities P_j and the standard activity times, to create instances with varying unequal workload distributions.

We introduce the use of the Theil index [18] as used in economics, to systematically investigate scheduling performance for various settings of relative workload. In our methodology, the Theil index expresses the inequality of the workload distribution. The Theil index is a value between 0 and log(m), which is calculated based on the individual workloads of the resources with the formula:

$$T = \frac{1}{m} \sum_j \frac{w_j}{\bar{w}} * \log \frac{w_j}{\bar{w}}, \tag{1}$$

where m is the number of resources, w_j the workload of resource j and $\bar{w}$ the average workload of all resources. This value is a measure of entropy; equal workload corresponds to values near 0, very unequal distributions go towards $\ln(m)$. In Fig. 3 we show three distributions of workload of resources for different Theil index values.

In our experiments, we generate workload distribution instances to sample configurations with different Theil index values. We present our results averaged over a large number of runs with varying Theil index values. The values P_j determine the skewness of the workload. They are generated using two functions: an exponential function of the form $P_j = \beta(\alpha^j / \sum_m \alpha^m)$, with α to vary the skewness, and β to vary the sum over P_j; a step function, where $\gamma(\leq m)$ number of resources resources have a high P_j of $(1 + \delta) * (\beta/(m + \gamma * \delta))$, and $m - \gamma$ number of resources have a low P_j of $(\beta/(m + \gamma * \delta))$, with δ to vary the high relative workloads.

4.3 Benchmark

We benchmark the performance of MPAEX relative to centralized and static schedule optimization methods. The semi-dynamical hospital patient scheduling model of Sect. 4.1 allows us to explicitly compare and contrast with more traditional centralized techniques. Importantly, these centralized heuristics do not have the restriction that patients must be scheduled in order of arrival, nor that the final schedule must be a Pareto improvement considering all patients.

Scheduling problems can be solved with exact solvers or heuristics [20]. Exact solutions like Branch and Bound (B&B) [21] require exponential time and solving large instances (like 200 patients on 8 resources) optimally is intractable. In practice, we could only run B&B on small problems: i.e., up to 10 patients on 4 resources.

Since we are interested in problems of more realistic size, we turn to well established scheduling heuristics to get good performance in reasonable time. From the literature, we used three centralized heuristics: two local search algorithms: a hill climber (HC) and simulated annealing (SA) [22], and least increment dispatching (LI) where priority is based on least increment in the overall objective [23]. All three heuristics (LI, HC, SA) need centralized information. Although that means they are not algorithms applicable in practice, we used them as a comparison for our distributed approach. In our experiments, SA and LI perform very close to each other for all instances, and better than our HC approach. Because of this, and the reasonable running time of LI (compared to SA) we only compare to LI in our results.

For instances of the equal activity times scheme, as defined in Sect. 4.2, we can calculate a lower bound for the optimal summed completion times of a schedule instance. To do so, we reduce each partial plan PP_i, each with varying number of activites $\{a_{i0}, \ldots, a_{ij}\}$, to a partial plan PP_i^* with exactly one activity $a_{ik} \in \{a_{i0}, \ldots, a_{ij}\}$, where a_{ik} is the activity in PP_i that requires the resource with the highest relative workload relative to the other resources required by PP_i. Determining the optimal summed completion of all PP_i^* is then easily computed as

$$\mathrm{LB}_{\mathrm{equal}} = \sum_{m} \sum_{l=1}^{l=M_m} \frac{1}{2} l(l+1),$$

with m a resource, and M_m the number of activities after reduction that are to be carried out on resource m. This value is a lower bound the for summed completion time of all PP_i (original), because changing the PP_i^*'s back into PP_i's by adding activities will never reduce the objective.

5 Experiments

In this section, we present the results of computer experiments to evaluate the performance of our distributed MPAEX approach. We compare the scheduling performance of MPAEX to the centralized static heuristic LI described in Sect. 4.3 for a wide range of settings. Additionally, we compare the performance of LI and the calculated lower bound of Sect. 4.3.

In the experiments, we use MPAEX for schedule improvement, with FCFS as well as with FCRS for initial scheduling. This gives us four distributed approaches [FCFS, FCRS, MPAEX(FCFS), MPAEX(FCRS)] and one centralized heuristic (LI) as benchmark. We let each of these five algorithms solve the same generated hospital patient scheduling instance, for a large number of scheduling instances. The overall objective is to minimize the unweighted sum of all individual patients' objectives (min $\sum_i C_i$). We measure performance for each instance by comparing the final schedule (σ^{dapp}) of the four distributed approaches, with the final schedule (σ^{LI}) from the centralized heuristic (LI). The performances of the approaches are presented relative to LI, computed as $\sum_i C_i(\sigma^{\mathrm{LI}}) / \sum_i C_i(\sigma^{\mathrm{dapp}})$ per instance and averaged over 50 instances for each presented data point. Thus, higher values represent better performance, and LI has a value of 1.

We calculate instances of the hospital patient schedule problem as developed in Sect. 4.1. The inequality of the relative resource workloads is characterized by the Theil index as discussed in Sect. 4.2. In each instance 200 patients, with an average of 2.5 activities per patient, need to be scheduled on eight resources. We motivate these numbers from the cases of Sect. 2.3: patients are usually issued a partial plan with a small number of activities selected from a limited set of resources. Furthermore, we will present robustness results for variations on this setting.

5.1 Relative workload

We present results for different Theil index values, for each of the four activity time schemes. We vary the Theil index of problem instances by setting the values of partial plan probabilities P_j, while keeping the average number of activities per patient constant (see Sect. 4.2, we use as sampling parameters $\beta = 2.5$, $1 \leq \alpha \leq 1.6$, $1 \leq \gamma \leq 8$, $2 \leq \delta \leq 6$). We group the instances created into different Theil index ranges (six equal ranges between 0 and 0.6) and the average result and variance per Theil index range (50 instances) is presented. We present the results in Fig. 4. The four graphs on the left present the performances of MPAEX(FCRS) and FCRS relative to LI (performance of 1), and similarly for MPAEX(FCFS) and FCFS and the right. The respective left

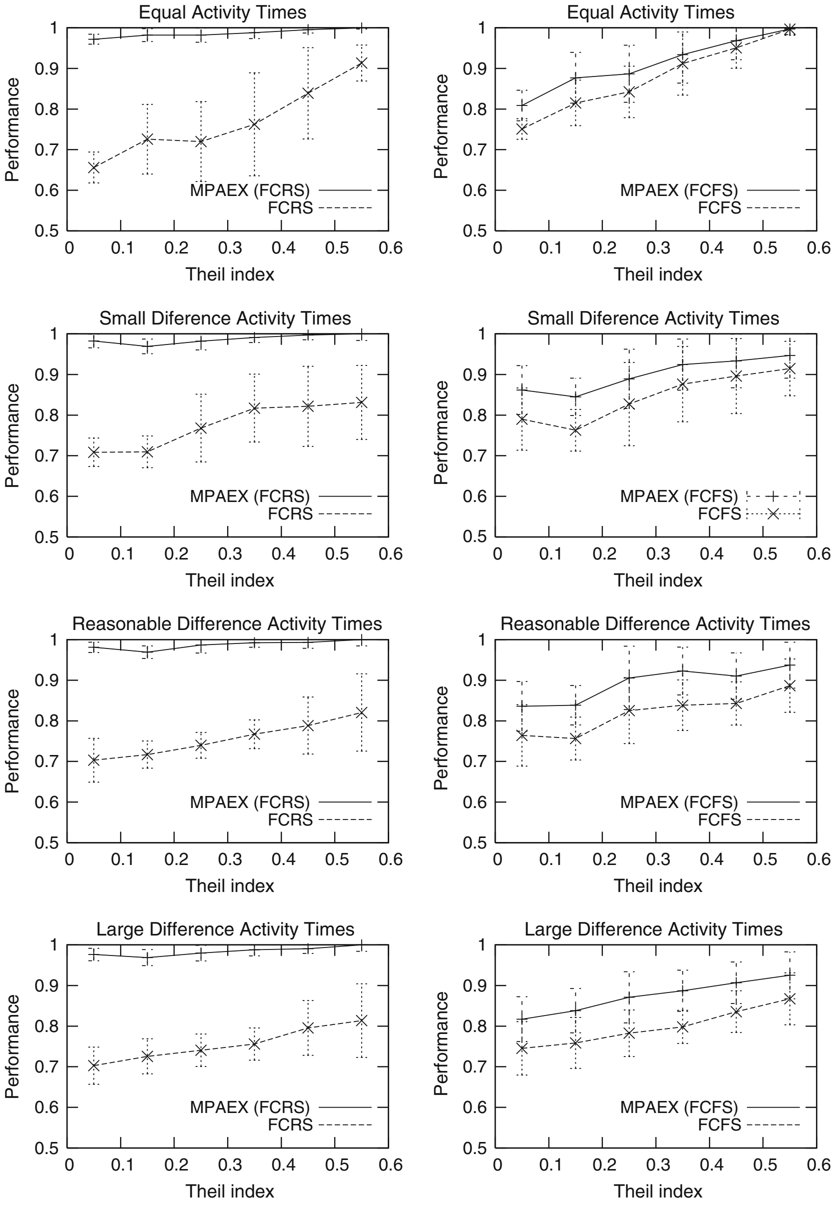

Fig. 4 Performances for different Theil index ranges relative to LI (performance of 1). *Error bars* indicate one standard deviation on either side of the calculated average

and right graphs can be compared directly, as the performances are compared against the same LI benchmark values (the separation between left and right graphs is strictly for presentation purposes).

From our results, we observe that the relative performances are robust for a wide range of problem settings. In all settings, our distributed MPAEX(FCRS) approach performs very close (between 96 and 100%) to LI, which needs centralized information. These results show that bilateral Pareto improvements can achieve very high scheduling performance. Furthermore, MPAEX(FCRS) performs much better than FCFS, where each patient in turn is scheduled optimally.

The results also show that the quality of the final schedule obtained by MPAEX is dependent on the initial scheduling: MPAEX(FCFS) has a performance lower than MPAEX (FCRS) and LI. However, MPAEX(FCFS) still improves on FCFS. The initial scheduling method FCRS has a performance lower than that of FCFS.

5.2 Lower bound

Although the computation of optimal schedules is intractable, we can compare the effectiveness of our benchmark LI to lower bound LB$_{\mathrm{e}}$qual for the specific equal activity times scheme developed in Sect. 4.2. Averaged over instances with different Theil index values, we find that LB$_e$qual has on average a value at 95±4% of the objective value of the schedule of LI. For settings with the equal activity times scheme, LI clearly performs close to the optimal performance.

5.3 Setting variations

We find that the performance of MPAEX scales well for the number of patients. We could only compare instances with less than 400 patients to LI, because of the long running time of LI. We therefore use the lower bound calculation with the equal activity times scheme, as a optimal performance (value of 1), and present performances relative to this value in Fig. 5. Relative performances are also robust in the other standard activity time schemes.

We also find that for a higher average of number of activities per patient, MPAEX performs even better. By varying the sum of the probabilities P_j, see Sect. 4.2, we vary the average number of activites per patient. The created instances are grouped into different ranges: given our setting of eight resources, we define seven equal ranges between 1 and 8. A value between 7 and 8 means that almost all patients have activities on all resources. The average result per range (50 instances) is presented in Fig. 6.

As the number of activities increases, MPAEX performs even slightly better than our centralized LI. All approaches, except FCRS, achieve similar performances for higher average number of activities per patient. The low performance of FCRS is caused by the fact if the number of activities increases the chance for a patient to get all his activities on only early timeslots decreases dramatically. With FCFS this chance is only dependent on the order of arrival, and not on the number of activities.

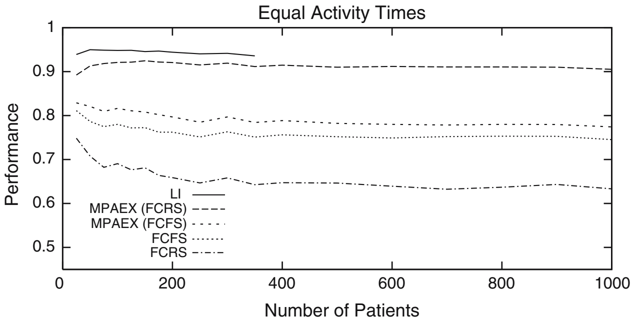

Fig. 5 Performances for varying number of patients, relative to LB$_{\text{equal}}$ calculation (performance of 1)

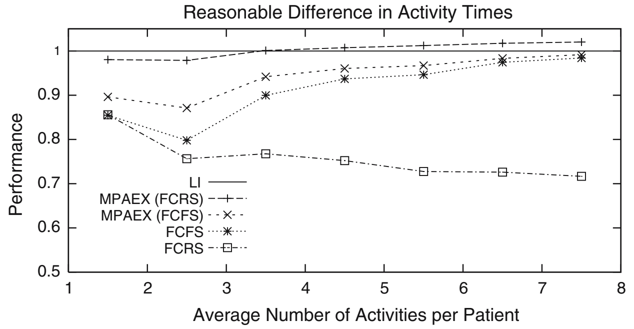

Fig. 6 Performances for varying average number of activities per patient (grouped) relative to LI (performance of 1)

6 Discussion

As in general queuing theory, due to the stochastic nature of actual hospital scheduling practice, short waiting times has to be traded-off against efficient resource usage [24]. In our model, this is not an issue because the resources are always efficiently used. Instead, we recognize that there is a secondary efficiency measure in terms of waiting time for patients with multiple appointments. By using a rescheduling approach, like MPAEX, efficient resource usage can be achieved with the initial scheduling method, while patient waiting times are improved by rescheduling.

Waiting time, as we defined it, is of course not necessarily the single important issue for the patient. For some patients the certainty of a fixed appointment is of greater value than a better schedule, although rescheduling might be necessary for medical reasons. Other patients may be more flexible, and guaranteeing not-worse rescheduling can be very attractive to them. In practice, patients could express whether they would like to be considered for rescheduling.

The scheduling problem we consider in this paper is a type of "open shop" scheduling problem from the traditional OR literature [20]. In open shop problems, all jobs (here patients' partial plans) consists of as many activities as there are resources. Our patient scheduling problem therefore corresponds to an open shop problem with processing times including values of zero: $O|p_{ij} = \{0, \text{sat}_{m_{ij}}\}|\sum C_i$ (standard scheduling notation), with p_{ij} the processing time of activity a_{ij}, and sat$_{m_{ij}}$ is the standard activity time of resource m_{ij}. The general open shop problem is NP-hard [25]. Different to most OR approaches, we do not consider performance guarantees, but average performances, given the stochastic parameters and for large number of patients.

Compared to FCFS and FCRS, one would wonder whether more efficient initial schedules could be achieved, of quality comparable to the rescheduling results presented here. In related work [26] we investigate efficient methods for online initial scheduling. Approaches for effective scheduling solutions we explore include adaptive methods of resource usage, and multilateral coordination between resources. In all, we find online initial scheduling to be a very complex problem, where successful approaches depend considerably on the particular resource characteristics and patient properties. In future research, we aim to integrate the two approaches to obtain both highly efficient schedules and reduce the number of appointments that need rescheduling, thus improving patient service level.

7 Conclusions

We have presented a multi-agent Pareto appointment exchanging algorithm, MPAEX, as a robust, dynamic and distributed solution for patient activity scheduling and rescheduling that actively exploits the Pareto improving scheduling opportunities that are present in hospital patient schedules where patients undergo multiple activities.

Models of the hospital patient scheduling problem were discussed in terms of the "health care cycle" where a doctor repeatedly orders sets of activities to diagnose and/or treat a patient. Additionally, we presented two models for the current practice of initial patient scheduling.

We introduced the Theil index to the hospital patient scheduling domain, to capture the degree of inequality in terms of relative workload between resources that are needed for patient scheduling. In this manner, we demonstrate how a broad range of possible scheduling problems with different relative workloads can be generated.

To compare against existing scheduling solutions, we presented a stylized semi-dynamical version of the actual dynamic problem: first, we schedule patients in order, and then we improve on that schedule with MPAEX.

In experiments over a broad range of such semi-dynamic hospital patient scheduling problems, we show that the MPAEX algorithm arrives at scheduling solutions that are almost as good as a centralized benchmark. Furthermore, by use of a calculated lower bound, we show that performances are close to the optimum. The high performance of MPAEX is robust for variations in the model settings.

Unlike centralized heuristics for solving scheduling problems, our multi agent approach can straightforwardly be used in a dynamic environment. Agents can interact concurrently, with local dynamic information (such as cancellations, disruptions, expired resource constraints).

In health care preferences on resource utilization (such as rosters, staff preferences) and of patients (combination appointments, online patient calendars) are inherently distributed. Multi-agent systems can capture such distributed preferences: agents try to optimize the schedule for their owner according to the preferences within the rules of scheduling. They will not accept a schedule that is worse than its current one. This is of great value to get collaboration of patients and system acceptance.

We have shown results on a semi-dynamic model to provide fundamental insights in the character of hospital patient scheduling and the performance of decentralized dynamic approaches versus traditional centralized static approaches. The setup we developed also provides a proper setting for future approaches and solutions. On the other hand, it is clear that real life problems are fully dynamic. Modeling such fully dynamic scheduling problems involve many more parameters that need to be set and fit to an actual case. In practice, much care has to be taken to obtain generic results from such fully dynamic models. In future research, we will develop dynamic cases in our studies in actual hospitals.

Acknowledgements This research is part of ToKeN (NWO grant number 634.000.021).

References

1. Vermeulen I, Bohte S, Somefun K, La Poutré H (2007) Improving patient activity schedules by multi-agent pareto appointment exchanging. In: Proceedings of the IEEE international conference on E-commerce technology, CEC/EEE, pp 56–63
2. Vissers JM (1998) Patient flow-based allocation of inpatient resources: a case study. Eur J Oper Res 105:356–370
3. Marinagi C, Spyropoulos CD, Papatheodorou C, Kokkotos S (2000) Continual planning and scheduling for managing patient tests in hospital laboratories. Artif Intell Med 20(2):139–154
4. Spyropoulos CD (2000) Ai planning and scheduling in the medical hospital environment. Artif Intell Med 20(2):101–111
5. Paulussen TO, Jennings NR, Decker K, Heinzl A (2003) Distributed patient scheduling in hospitals. In: Proceedings 18th international joint conference on AI
6. Nealon J, Moreno A (2003) Agent-based applications in health care. In: Nealon J, Moreno A (eds) Applications of software agent technology in the health care domain. Birkhueser, pp 3–18
7. Weiss G (ed) (1999) Multiagent systems: a modern approach to distributed artificial intelligence. MIT Press, Cambridge, MA
8. Decker K, Li J (2000) Coordinating mutually exclusive resources using gpgp. Auton Agent Multi Agent Syst 3(2):133–157
9. Hannebauer M, Müller S (2001) Distributed constraint optimization for medical appointment scheduling. In: AGENTS '01: Proceedings of the fifth international conference on autonomous agents, ACM Press, New York, pp 139–140
10. Wellman M, Walsh W, Wurman P, MacKie-Mason J (2001) Auction protocols for decentralized scheduling. Games Econ Behav 35:271–303
11. Reeves DM, Wellman MP, MacKie-Mason JK, Osepayshvili A (2005) Exploring bidding strategies for market-based scheduling. Decis Support Syst 39(1):67–85
12. Sen S, Durfee E (1996) A contracting model for flexible distributed scheduling. Ann Oper Res 65:195–222
13. Czap H, Becker M (2003) Multi-agent systems and microeconomic theory: a negotiation approach to solve scheduling problems in high dynamic environments. In: Proceedings of 36th annual Hawaii international conference on system sciences, pp 83
14. Maruster L, Weijters T, de Vries G, van den Bosch A, Daelemans W (2002) Logistic-based patient grouping for multi-disciplinary treatment. Artificial Intelligence Med 26(1-2):87–107
15. Mas-Colell A, Whinston M, Green JR (1995) Microeconomic theory. Oxford University Press, Oxford
16. Chevaleyre Y, Dunne PE, Endriss U, Lang J, tre ML, Maudet N, Padget J, Phelps S, guez Aguilar JAR, Sousa P (2006) Issues in multiagent resource allocation. Informatica 30:3–31
17. Endriss U, Maudet N, Sadri F, Toni F (2006) Negotiating socially optimal allocations of resources. J Artif Intell Res 25:315–348

18. Theil H (1967) Economics and information theory. North-Holland, Amsterdam
19. Sgall J (1996) On-line scheduling. In: Fiat A, Woeginger GJ (eds) Online algorithms. Springer, Berlin, pp 196–231
20. Brucker P (2001) Scheduling algorithms. Springer, Berlin
21. Land AH, Doig AG (1960) An automatic method for solving discrete programming problems. Econometrica 28:497–520
22. Aarts E, Lenstra J (2003) Local Search in Combinatorial Optimization. Princeton University Press, NJ
23. Haupt R (1989) A survey of priority rule-based scheduling. OR Spect 11(1):3–16
24. Porter ME, Teisberg EO (2004) Redefining competition in health care. Harv Bus Rev 82:64–76
25. Hoogeveen H, Schuurman P, Woeginger GJ (1998) Non-approximability results for scheduling problems with minsum criteria. In: Proceedings 6th international integer programming and combinatorial optimization conference. Lecture notes in computer science, vol 1412. Springer, Berlin, pp 353–366
26. Vermeulen I, Bohte S, Elkhuizen S, Lameris J, Bakker P, La Poutré H (2007) Adaptive optimization of hospital resource calendars. In: Proceedings of the 11th conference on artificial intelligence in medicine, AIME 07. Lecture notes in computer science. Springer, Berlin