

Distributed Decision Making in Hospital Wide Nurse Rostering Problems

Stefaan Haspeslagh, Patrick De Causmaecker

Katholieke Universiteit Leuven Campus Kortrijk, E. Sabbelaan, 8500 Kortrijk, Belgium,
 {stefaan.haspeslagh,Patrick.DeCausmaecker}@kuleuven-kortrijk.be

Greet Vanden Berghe

KaHo St.-Lieven, Information Technology, Gebr. Desmetstraat 1, 9000 Gent, Belgium,
 Greet.VandenBerghe@kahosl.be

This contribution considers the distributed version of the Nurse Rostering Problem. A hospital consists of several wards that have a high degree of autonomy. Each ward maintains its local roster. This local decision making problem is of combinatorial complexity. By sharing and exchanging resources, the wards want to improve their rosters and tackle staff shortages. In this paper, a negotiation based model is studied in which hospital wards are represented by agents. The agents negotiate about their rosters: in case of staff shortage, requests are raised by the agents. An experimental setup is provided as proof of concept.

Keywords: Real World Scheduling, Rostering, Agent Based Scheduling.

1 Introduction

The purpose of this paper is to study the possibilities of a distributed approach in the case of the nurse rostering problem. We envisage a system consisting of units (wards) with a large degree of autonomy and highly detailed local decision making. The independence of these units is limited by the sharing of resources and by the opportunities or obligations for inter unit assistance. This assistance may increase the efficiency of one unit while only slightly hampering the performance of another one. On the long term, the performance of the larger systems should benefit. So, in its simplest form, we must consider two levels of decision making which we label local and global. The local system takes the highest level of detail into account. We are especially interested in the case where the local decision problem is of a combinatorial complexity. Various local details are aggregated and translated into variables that can be handled at the global level, which, at the same time, introduces supplementary requirements, constraints and preferences. The decision problem at this global level may be of a combinatorial complexity as well.

A system consisting of autonomous units naturally leads to an agent based model. Such models have been intensively studied over the last decade, e.g. by Valckenaers et al. [11]. Rather recently the link with optimisation has obtained some attention. Shen et al. [8] give an updated review in the field of intelligent manufacturing. We think that the optimising behaviour of an agent based system should be studied as close to the real world problems as possible. In this paper we study the behavior of a system of negotiating agents for a distributed nurse rostering problem. We select a specific negotiation protocol for the global problem and an algorithm for the local problem. Our aim is to demonstrate the feasibility of such an approach and to identify some relevant questions for further research. Our ultimate aim is to demonstrate the feasibility of such an approach. In this paper, we discuss requirements, propose a model and identify some relevant questions for further research.

The Nurse Rostering Problem (NRP) has been the subject of intensive study. An overview is presented by Burke et al. [4]. Typically a hospital is divided into several wards, and rosters

are generated at ward level. Rosters have to satisfy each nurse's work regulation while realizing a given coverage. Nurse preferences must be satisfied if possible. The criteria and objectives used to evaluate rosters at ward level are different from those at hospital level.

At the level of an individual ward, the criteria are those that have been analysed in previous work [4]. Demands stem from the workload per time unit or shift in the ward and are typically expressed as a number of nurses of a specific qualification. Work regulations define, sometimes nurse-specific, values for e.g. the maximum number of consecutive night shifts, the minimum and maximum number of consecutive days off, and so on. In addition to those constraints, one wants to take personal preferences and ad hoc requests of nurses into account.

At the hospital level other criteria are to be considered. Here, the manager wants to guard the fairness of the work load distribution, wants to raise certain quality levels, wants to minimize the personnel cost, wants to decide where resource shortages are allowable at peak moments . . .

As an example of a local problem that may be resolved at the global level, we will consider the cases of sudden staff shortage due to absence or unexpected overload. These may be hard to deal with at the local level where a longer term schedule should not be disturbed too drastically, while chances are higher that a peer ward accidentally has some spare capacity at or about this specific moment in time. Trivedi and Warner [10] and Gascon et al. [7] introduce a pool of float nurses. They solve shortages by using nurses from this pool. Siferd and Benton [9] tackle shortages by calling nurses from other wards and by allowing nurses to work overtime. Meisels and Kaplanski [6] pioneered a negotiation based approach. These authors introduce three stages. In stage 1, scheduling agents generate a local solution. In stage 2, a global feasible solution is constructed. The last stage is used by the agents to improve their local schedules. Meisels and Kaplanski assume a fixed pool of shared resources. Bard and Purnomo [2] propose various ways of tackling staff shortages. Relevant for the present discussion is the one in which initially negotiation takes place with a pool of floating nurses. Consequently, nurses who are not needed by the wards, are placed on an on-call list. If none of the proposed solutions suffices the hospital can call in agency nurses.

In this contribution we will use a negotiation based model for the optimisation at the global, hospital wide, level. In case of shortage, we allow for requests to be raised by wards. Other wards of the hospital consider re-rostering in order to satisfy the demand and post offers to resolve the request. The model is based on an extension of the Contract Net Protocol.

In Section 2, the model for the nurse rostering problem at the level of a ward is presented. Section 3 presents the negotiation model. In Section 4 we elaborate on the required components and their specifications as they follow from the discussions in the previous sections. Section 5 discusses the experiments we performed as a proof of concept. Section 6 draws conclusions and describes future research.

2 The nurse rostering problem for one ward

In [4] a large number of models and approaches are listed and discussed. Most authors treat the Nurse Rostering Problem as a matching exercise between the needs of the hospital and the constraints (legal and personal) under which the ward personnel operates. We use the model of [3] where coverage demands, in the simplest case, are considered hard and personnel rules and preferences are implemented through a penalty function weighting the importance of a specific constraint. As an extension, coverage demands can be considered soft as well, if desired. This complicates the optimisation procedure but may offer a richer set of options to be negotiated among agents. Figure 1 shows a common used representation for rosters. The columns stand for

the days of the planning period. The rows consist of the employees working in the ward. Every working day consist of a number of shift types (night, morning, late, ...).

	Mo	Tu	We	Th	Fr	Sa	Su
Q1	N	N		M	M	L	
Q2	L	M	M			N	N
Q3	L	L	L	L	L		M
Q4					L	L	L
Q5	M	M	N	N		M	M

Figure 1: Example roster

3 The negotiation model

The negotiation model we use in our current approach is based on an extension of the Contract Net Protocol [1] that is called the Extended Contract Net Protocol (ECNP). It is designed for use with cooperative and self-interested agents. In this protocol, managers make calls for bids. Contractors try to formulate appropriate bids to those calls. In the ECNP, managers can concurrently call for bids and a contractor agent can concurrently manage several negotiation processes with multiple managers. The protocol is more efficient in time and is fault tolerant.

From our point of view, wards can be manager and contractor at the same time. In case of a staff shortage, a ward will raise a request. A request can either be a specific call for a nurse to work a particular shift or a request to make a (partial) roster of an employee lighter. Other wards consider re-rostering in order to answer the call. Meanwhile they raise requests for their own problems. First we explain the negotiation protocol in more detail. Second, we elaborate on the benefits of this protocol. Last, we discuss the requirements that this protocol imposes.

The parties in the protocol are the the manager and the contractor. The manager is the ward that raises requests, the contractors make offers to those requests. Negotiation is carried out in five steps:

1. an announcement is made by the manager to all the possible contractors of which it believes that they are capable to solve his problem.
2. The contractors formulate a proposal. They send a PreBid message to the manager. The manager compares the received PreBids and sends a PreAccept message to the contractor that made the best offer. The other wards receive a PreReject message.
3. A contractor that receives a PreReject may trie to make a better proposal and sends a new PreBid. If a manager receives a better offer than the best offer received so far, it sends a PreAccept message to the ward of the new best bid and a PreReject to the ward of the previous best offer.
4. After a period of time, the contractors send a DefinitiveBid to the manager.
5. The manager decides which definitive offer is best and a DefiniveAccept message is sent to the ward offering the bid. A DefinitiveReject message is sent to the others.

The advantages of this extended version of the Contract Net Protocol are fourfold:

- The protocol itself gives contractors the possibility to run several negotiation processes in parallel.
- It follows that the accumulated length of all the negotiation processes between the agents is reduced. In the original protocol, the only way for the managers to reduce the length of their negotiations, was through managing parallel negotiations.
- By introducing the PreBidding phase, the protocol reduces the number of de-commitments between the contractors.
- It avoids penalising contractors that break their commitments.

The above motivations state the appropriateness of the protocol for this application. In order to process the requests, a simulation engine has to be designed. The use of the presented protocol imposes some demands on the simulation engine. The engine must be capable of simulating local changes to the rosters. A ward manager does not want to totally change the rosters in order to satisfy demands from other wards. Also, if a bid is rejected, a small modification of the bid could be sufficient to satisfy the request.

4 Application design, requirements and specifications

This section gives an overview of the agents and software modules used in our application. Every ward is represented by four agents. A ManagerAgent represents the role of the manager in the previously discussed ECNP. Analogously, the role of the contractor is taken by the ContractorAgent. The SimulationAgent calculates and processes bids. The ProblemIdentificationAgent is responsible for raising requests. We elaborate on those four agents in more detail below. Some software modules, e.g. a communication module, needed by the agents, are provided as well. We discuss only one software module, the SimulationEngine. Figure 2 illustrates the ward design. For each agent, we state its task, identify the other agents it cooperates with and we list the software modules used by the agent:

- ManagerAgent:

The task of this agent is to raise requests and choose between the various reply offers made by the contractors to those requests. The agent takes several considerations into account when formulating requests. For example, it may decide to only send the request to the wards it believes are capable of producing a positive answer to its question. E.g. the Intensive Care Unit and the Coronary Care Unit may be more likely to exchange nurses because they employ similarly skilled people. Sending the same request to the geriatrics ward might be without purpose. The ManagerAgent cooperates with the ProblemIdentificationAgent to formulate questions and with the SimulationAgent to evaluate bids. The CommunicationModule is used to perform the actual communication between the wards.

- ContractorAgent:

The task of the ContractorAgent is to calculate PreBids as an answer to calls for bids from the managers. Furthermore, if a PreReject message is received, the ContractorAgent needs to adjust its offer and formulate a new one. The Contractor makes decisions according to its current views. For example, for friendly wards, the contractor may decide to make offers

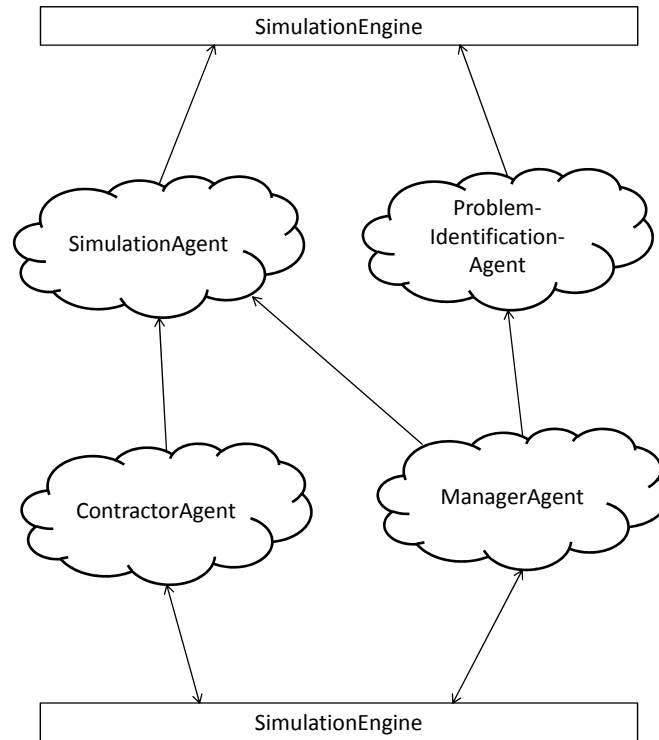


Figure 2: Ward software design

that have a more significant impact on the quality of its roster, while for other wards it wants to be more conservative. The SimulationAgent helps the contractor to calculate bids. Communication is handled by the CommunicationModule.

- SimulationAgent:

The role of the SimulationAgent is to help the ContractorAgent with the creation of offers and to help the ManagerAgent in evaluating the bids it receives. In its simplest form, the role of this agent could be supported by the other agents. A more advanced version should be able to learn which offers and questions the wards are most responsive to. If the SimulationAgent detects that certain questions are never answered and that certain offers are always rejected, it should stop to raise such questions and other offers must be created. Therefore, a dedicated agent is designed. The SimulationAgent carries the beliefs and the decision making strategies of the ward manager. The SimulationEngine module performs the actual simulation. In a specific implementation, several instances of these agents and modules may be present in order to avoid bottlenecks. If this is the case, communication between the agent instances will be necessary to support the learning behaviour.

- ProblemIdentificationAgent:

The ProblemIdentificationAgent's task is to identify problems occurring in the ward's rosters. Again, the ProblemIdentificationAgent represents the beliefs of the ward manager. Wards may differ w.r.t. their conception of when a roster is problematic. The ProblemIdentificationAgent makes use of the same SimulationEngine as the SimulationAgent, to identify problems.

The SimulationEngine allows to access the local schedule and the local planner. As stated before, the requirements of the SimulationEngine stem from the negotiation protocol used. Local changes of a ward's roster are necessary for the protocol to work. We explain what we mean by local changes. We identify three requirements:

1. The time period in which planning occurs must be adjustable. This is represented by the vertical rectangle in figure 3.1. When replanning to fit a particular shift in the roster, the ward manager does not want the entire planning period to change. Only small changes, for instance, within a time period of a couple of days before or after the time where the shift must be covered, are allowed.
2. The SimulationEngine must be capable of rostering only for a specific number of employees. The horizontal rectangle in figure 3.2 represents this requirement. Consider for example, that a question for a head nurse is raised. In this case, the rescheduling must be restricted to the rosters of the nurses who have an equivalent skill. After all, nurses with lower skills are not qualified to cover such shifts.

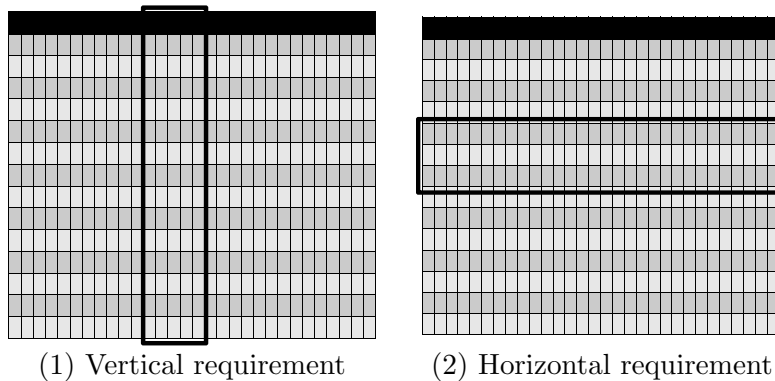


Figure 3: SimulationEngine requirements

3. Combinations of the first two requirements. In this way, small regions in the ward's roster can be selected for rescheduling. For example, one can state that rostering only has to be carried out for two head nurses, for one day before and one day after a particular shift. Figure 4 shows a graphical representation of this requirement.

An implementation of the SimulationEngine needs to map these requirements to the local planner's model.

5 Implementation and experimental setup

Up until now, we implemented a simplified version of the application described above. We briefly discuss the main simplifications made. The SimulationEngine does not yet comply to the requirements that we mentioned above. This is due to shortcomings in the local planner we currently use. The local planner can only reschedule an entire roster. This imposes limitations to the SimulationAgent and the ProblemIdentificionAgent. E.g., for now, the ProblemIdentificionAgent searches for the employee with the worst roster. The ManagerAgent uses that roster to raise a request. For handling requests, the SimulationAgent tries to fit every shift of the problematic roster into its ward's roster. A shift can be covered if the total penalty of the roster does not increase by some

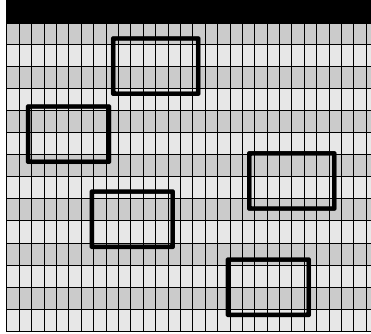


Figure 4: Combination requirement

percentage. Currently, the ManagerAgent and ContractorAgent do not build long term relations with wards. Every ward is considered equal and every request is sent to every ward.

We set up an experiment with five wards. The data for each ward's roster and the local planner is found at [5]. The wards negotiate on their rosters in order to obtain a better global solution.

6 Conclusion and future research

With the experiments described above, we want to deliver a proof of concept. However, the current experimental setup is limited. The number of possible personnel exchanges, handled by negotiation, is far less than the number of solutions considered by a local solution method. This must have its consequences for the complexity that can be handled at both levels. The present model does not yet take this into account. Limitations of the implementation are the following:

1. the agents do not keep track of the history. E.g., both the ManagerAgent and the ContractorAgent should use the information learned from previous negotiations for further negotiation. They can use that information to build a list of friendly wards. Those friendly wards can then be favoured in future negotiations. The SimulationAgent could maintain a list of possible answers to requests and could make a classification of those answers. Analogously, the ProblemIdentificationAgent can maintain a set of frequently occurring problems and the possible request associated with those problems.
2. the local planner that we currently use, is too limited in functionality. While discussing the SimulationEngine, we elaborated on the requirements in functionality of the local planning engine. Our current planner does not comply to those requirements.
3. the expressive power of the local nurse rostering model. For instance, for friendly wards, the ContractorAgent may consider offers with a significant impact on the ward's penalty. This aspect is not incorporated into the current objective function.

In future research, the influence of the negotiation protocol on the SimulationEngine's requirements, needs to be investigated. We previously identified the requirements imposed by the Extended Contract Net Protocol. Analogous work needs to be done with other negotiation models. Next, the results obtained by negotiating according to those protocols, have to be compared. Then, a comparison with results from central approaches is necessary.

References

- [1] S. Aknine, S. Pinson, and M. F. Shakun (2004), An extended multi-agent negotiation protocol, *Autonomous Agents and Multi-Agent Systems* **8**(1), 5 – 45.
- [2] J. Bard and H. Purnomo (2004), Real-time scheduling for nurses in response to demand fluctuations and personnel shortages, In *Proceedings of the fifth International Conference in the Practice and Theory of Automated Timetabling, Pittsburgh*, September 6-10, 67 – 87.
- [3] E. K. Burke, P. Cowling, P. De Causmaecker, and G. Vanden Berghe (2001), A memetic approach to the nurse rostering problem. *Applied Intelligence* **15**(3), 199 – 214.
- [4] E. K. Burke, P. De Causmaecker, G. Vanden Berghe, and H. Van Landeghem (2004), The State of the Art of Nurse Rostering. *Journal Of Scheduling*, **7**(6), 441 – 499.
- [5] T. Curtois. Nurse rostering problems and benchmarks, <http://www.cs.nott.ac.uk/~tec/NRP/>.
- [6] E. Kaplansky and A. Meisels (to appear), Distributed personnel scheduling - negotiation among scheduling agents, *Annals of Operations Research*.
- [7] V. Gascon, S. Villeneuve, P. Michelon, and J. Ferland (2000), Scheduling the flying squad nurses of a hospital using a multi-objective programming model. *Annals of Operations Research* **96**(1), 149 – 166.
- [8] W. Shen, Q. Hao, H. J. Yoon, and D. H. Norrie (2006), Applications of agent-based systems in intelligent manufacturing: An updated review, *Advanced Engineering Informatics* **20**, 415 – 431.
- [9] S. Siferd and W. Benton (1992), Workforce staffing and scheduling: Hospital nursing specific models, *European Journal of Operations Research* **60**(3), 233 – 246.
- [10] V. Trivedi and M. Warner (1976), A branch and bound algorithm for optimum allocation of float nurses, *Management Science* **22**(9), 972 – 981.
- [11] P. Valckenaers, K. Hadeli, B. Saint Germain, P. Verstraete, and H. Van Brussel (2006), Emergent short-term forecasting through ant colony engineering in coordination and control systems, *Advanced Engineering Informatics* **20**(3), 261 – 278.