Adaptive Negotiation In Managing Wireless Sensor Networks

Thao P. Le, Timothy J. Norman, and Wamberto Vasconcelos

Department of Computer Science
King's college, University of Aberdeen, AB24 3UE, UK
{thao.le, t.j.norman}@abdn.ac.uk
wvasconcelos@acm.org

Abstract. The allocation of resources to tasks in an efficient manner is a key problem in computer science. One important application domain for solutions to this class of problem is the allocation of sensor resources for environmental monitoring, surveillance, or similar sensing tasks. In real-world problem domains, the problem is compounded by the fact that the number of tasks and resources change over time, the number of available resources is limited and tasks compete for resources. Thus, it is necessary for a practical allocation mechanism to have the flexibility to cope with dynamic environments, and to ensure that unfair advantages are not given to a subset of the tasks (say, because they arrived first). Typical contemporary approaches use agents to manage individual resources, and the allocation problem is modelled as a coordination problem. In existing approaches, however, the successful allocation of resources to a new task is strongly dependent upon the allocation of resources to existing tasks. In this paper we propose a novel negotiation mechanism for exchanging resources to accommodate the arrival of new tasks, dynamically re-arranging the resource allocation. We have shown, via a set of experiments, that our approach offers significantly better results when compared with an agent-based approach without resource re-allocation through concurrent negotiation.

1 Introduction

When a sensor network is deployed it is typically required to support multiple simultaneous tasks. A given sensor can provide different amounts of information to each individual task. Tasks are broken down as sub-tasks and can appear at any time placing varying demands on sensor resources. In such multiple-sensor and multiple-task problems in dynamic environments, conflicts between sub-tasks may occur for the use of the same sensor resource. Thus, efficient mechanisms to allocate individual sensors to appropriate sub-tasks on the basis of information need are necessary.

The resource-task allocation problem is at least as hard as the Knapsack problem which is NP-Complete [5]. In the current state of the art, there is no generally adopted approach to solve this class of problems, and researchers have made many assumptions in order to be able to provide a solution to a subset of the generic problem (e.g. considering only systems where sensors are identical,

sub-tasks are of the same type, or systems where sub-tasks require the exclusive use of sensor resources). In an attempt to relax such assumptions, we have focused on resource allocation problems in heterogeneous and dynamic sensor networks. Specifically, we employ an agent-based approach allowing sensors to be shared between sub-tasks. In so doing, however, the success of a sub-task strongly depends on the allocation of earlier sub-tasks. Moreover, in practical scenarios not all sub-tasks will operate in a cooperative manner (i.e. the agents coordinating the sub-tasks might not be willing to participate in the reassignment of sensors without compensation).

Negotiation techniques have long been used in multi-agent systems to resolve disagreements between agents to enable them to come to agreements that all parties can live with [10]. It is, therefore, appropriate to investigate the use of negotiation mechanisms for reassigning sensor resources. In doing this, we introduce another objective for agents: maximising profit. A task (represented by a buyer agent) in need of a particular sensor might be willing to give up part of its profit to a potential seller (representing another task) in exchange for the service of that sensor. If the seller can find an alternative sensor to replace that particular sensor, it will be beneficial to do the exchange if it is able to obtain additional profit from the buyer. For the buyer, it will have a chance of completing its allocation, thus achieving the objective and also obtaining a profit that is unavailable otherwise. We further demonstrate that it is advantageous for the buyer to have a number of such negotiations concurrently because this increases its chance of being successful.

In this paper, we make the following contributions to the state of the art. First, we enhance sensor-task allocation mechanisms by employing an adaptive negotiation mechanism in the allocation process. This makes our approach more applicable in realistic situations where sub-tasks compete for resources. Additionally, to the best of our knowledge, this presents the first model introducing negotiation as a post-processing step to improve the actual allocation process. Through simulations, we empirically demonstrate that our extended model provides an improvement in the number of completed tasks.

The remainder of this paper is organized as follows: Section 2 formulates the sensor-task allocation problem. Section 3 presents our agent-based approach and Section 4 extends this model by incorporating a novel negotiation mechanism specifically for resource exchange between self-interested task-agents. We present an in-depth analysis of our experimental results in Section 5, followed by Section 6 where we relate our model to existing research in this area, discuss the shortcomings of our model and point towards avenues for future research. Finally, Section 7 concludes.

2 Sensor-Task Allocation Problem

The problem considered in this paper involves allocating a collection of sensors to a number of tasks in order to satisfy the information requirements of those tasks.

A sensor s_i is defined as a tuple $\langle \gamma_i, l_i, r_i, c_i, u_i \rangle$ where $\gamma_i \in \Gamma$ specifies s_i's type (Γ is the set of all sensor types); l_i and r_i are the location and sensing range of s_i; c_i is the cost of using s_i; and u_i is the maximum utility s_i can provide in a single time unit.

Tasks may arrive at any time and may last for any duration. A task M is defined by a specific geographic location, starting time and duration. M is composed by a set of sub-tasks T. Each sub-task $t_j \in T$ has a specific type and is defined as a tuple $\langle l_j, r_j, d_j, p_j, b_j \rangle$ where l_j and r_j specifies t_j's location and operational range; d_j is the sensing demand that t_j requires; p_j is the profit t_j will achieve if successfully allocated; and lastly, b_j is the overall budget for the sub-task. The active time for t_j is within the duration of task M. We denote u_{ij} as the utility that s_i can provide to t_j, which is defined as a percentage of u_i calculated by the ratio between the overlap of the ranges of s_i and t_j and the range of s_i. If the operational areas of s_i and t_j do not intersect, the value of u_{ij} will be 0.

Given a set of available sensors $S = \{s_1, s_2, ..., s_n\}$ for t_j at t_j's starting time, we formulate the allocation for t_j as a mathematical programming problem. Specifically, an allocation to t_j is defined as the matrix $A_j = (x_{ij})_{n \times 1}$ where $x_{ij} = \{0, 1\}$ and $x_{ij} = 1$ denotes that sensor s_i is allocated to sub-task t_j. The utility that t_j achieves is calculated as: $U_{t_j} = \sum_{i=1}^{n} u_{ij} \times x_{ij}$. The cost of t_j's allocation is calculated as: $C_{t_j} = \sum_{i=1}^{n} c_i \times x_{ij}$.

An allocation A_j is valid if, and only if:

1. the total cost of an allocation must be within budget: $C_{t_j} \leq b_j$
2. the utility achieved must greater than or equal to the sensing demand (within a threshold ξ) for t_j: $U_{t_j} \geq \xi \times d_j$,
3. the set of sensor types of the sensors allocated to t_j must cover its information requirements: for all required type $\gamma_k \exists s_i : x_{ij} = 1, \gamma_i = \gamma_k$
4. sensors cannot be allocated to more than one type of sub-task at the same time (i.e. the only permit sensors to be shared between sub-tasks of the same type): $\sum_{j \in \overline{T}} x_{ij} \leq 1$ for all set $\overline{T}$ of sub-tasks with different types.

If A_j is valid, the profit that t_j will receive is calculated as $P_{t_j} = min(U_{t_j}/d_j, 1) \times p_j$. Task M will have a successful allocation if all of its sub-tasks are satisfied (A_j is valid $\forall t_j \in T$). The profit that M receives in this case is $P_M = \sum_{t_j \in T} P_{t_j}$, $\forall t_j \in T$.

Formally, the allocation problem is defined as:

$$\begin{aligned} \text{max:} \quad & \text{count}(M),\ \Sigma P_M \\ \text{s.t.:} \quad & A_j \text{ is valid } \forall t_j \in T \end{aligned}$$

In other words, we aim to utilize the set of sensors to maximize the number of successful tasks as well as obtain as much profit as possible for such tasks (emphasizing the number of successful tasks).

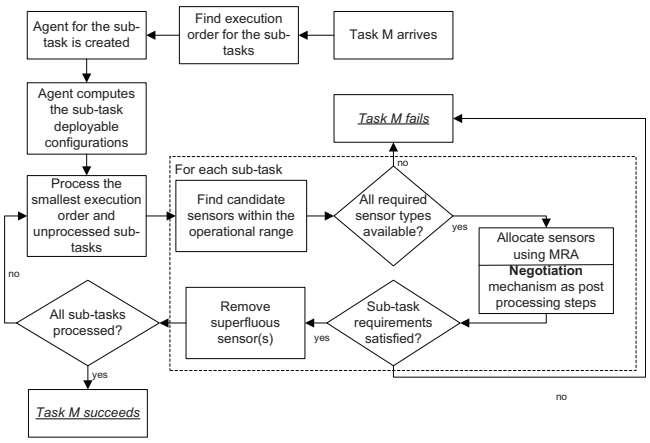

Fig. 1. Our proposed approach as a flowchart.

3 Agent-Based Sensor-Task Allocation

In this section, we present an approach to continuous resource allocation problem for sensor network management that offers significant efficiency improvements over existing solutions, while generating high quality solutions.

We assume that sensors of different types are deployed in an environment in a uniformly random manner, have varying sensing ranges and each sensor provides different utilities to different sub-tasks. The utility each sensor can contribute is computed by a predefined function for each task and depends on various factors such as sensor type, range, location and so on.

By a task, we mean a sensing task that requires information of a certain type, which may be contributed by one or more sensor types. Tasks can arrive at any time, and there may be more than one task active at any given time. Tasks may consist of a set of sub-tasks and each sub-task is defined by a specific location, operational range and type. Moreover, each task has a profit representing its importance, and this profit can only be achieved if the task is successfully allocated. Tasks also require different numbers of sensing resources (i.e. it has a sensing demand) and these requirements may not be met by a single sensor type. In such cases, different sensor types should be allocated together to meet the requirements of a sub-task. We use the term *Deployable Configuration* (DC) to refer to the set of resource types that an atomic task requires.

We propose a multi-agent system where each task is represented by a task agent. The task agent is responsible for the task. If a task is composed of sub-tasks, then that task's agent delegates those subtasks to other task agents. If a task agent represents an atomic task (i.e. the task have no sub-tasks), then the agent is only responsible for the determination and allocation of resources required to execute the task. In summary, agents of tasks are responsible for the delegation of subtasks to other agents while the agents of sub-tasks are responsible for the determination and allocation of resources.

Resource determination and allocation for each atomic task is managed by the agent of that task. Hence, for a composite task, overall resource determination and allocation is achieved in a decentralized manner by the agents representing atomic tasks within the composite task. The agents of the atomic tasks first determine the necessary resource types and then interact with the resources (sensors) on the area of their interest to allocate the necessary resources. In our approach, each sensor is represented by a sensor agent, knowledgeable about the location, range, type, battery life and utility of its sensor. Therefore, in order to allocate sensors for a specific atomic task, the agent of this task should interact with the sensor agents considering its requirements and constraints. Here, we assume that task agents compete for resources while sensor agents are purely cooperative.

As mentioned earlier, a task can arrive at any time and there may be more than one task active at any given time. When a new task T arrives, T is delegated to a task agent A_T (the sensor agent closest to the central of Ts range). A_T is responsible for controlling the process of finding an allocation for T as follows (see Figure 1):

1. Establish the execution order for sub-tasks. Basically, two tasks t_i and t_j belong to the same execution set (they can be executed at the same time) if their operational ranges do not intersect or their sensor type requirements do not overlap. However, if two tasks have the same type, both will be in the same execution set. Initially, the execution set containing t_0 will be processed first and followed by the set containing the next unprocessed task until all the tasks have been handled.
2. Delegate the sub-tasks (e.g., t_j) to task agents (e.g.,A_T^j).

A_T^j is knowledgeable about the constraints and requirements of the sub-task t_j. A_T^j computes the set of deployable configurations (DCs) for t_j. These DCs are determined by a semantic matchmaking process [13] and then used as the input for the actual allocation process. The key benefit in doing so is that the search space for finding the allocation solution can be greatly reduced (A_T^j only has an interest in sensors of a specific type if the deployable configurations of its sub-task contains this sensor type).

When a DC has been selected for t_j, the actual allocation steps are as follows:

The task agent A_T^j identifies candidate sensors within the operational range of t_j. A call for bids is issued to appropriate sensors. The call for bids includes information regarding its type, location, etc. Each sensor agent then makes an independent decision on whether and what to bid based on its type and workload. A response to a call will include the utility that can be provided and the cost associated with the use of this sensor.

Once bids are received, the coordinator agent attempts to allocate sensors to the sub-task using a multi-round allocation algorithm (MRA). MRA operates in the similar way to GAP-E algorithm [8]; typically, it is in the nature of this allocation algorithm that the various agent-based techniques differ. If A_T^j fails to satisfy its information requirements, it reports failure to the agent responsible

for A_T, and if the sub-task is critical to the overall task, all other task agents coordinating dependent tasks/sub-tasks will be requested to abort and release their resources. All sensor agents from which bids were received are informed of whether they are required.

In the MRA algorithm, sensors of various types are allocated to the sub-task in a number of rounds, one for each sensor type the sub-task requires. The first step is to set the order of selection of potential sensors using their priority. In this way, all sensors of the highest priority are considered first. Also, MRA introduces a budget (a constraint that governs the number of sensors that can be allocated to the atomic task) as part of its specification. From the bids received the allocation algorithm also has the costs associated with using specific sensors and the utilities they provide. The Fully Polynomial Time Approximation Scheme (FPTAS) algorithm which offers an approximation guarantee of $2 + \epsilon$ is then run with this as input along with an allocation from the remaining budget and utilities that sensors can provide to the task. This algorithm returns a revised allocation. If this allocation does not contain at least one sensor of the type being considered, the atomic task fails. Otherwise, the algorithm then reassesses the priority among sensor types (given the fact that sensors have been allocated) and proceeds to the next round if additional resources are required.

4 Negotiation for (Re-)Allocation of Resources

In this section, we detail our novel negotiation mechanism which can be used during the post-processing step in each round of the allocation algorithm outlined in the previous section. As has been argued, the problem inherent in a decentralised (or agent-based) approaches to the sensor-task allocation problem is that the order of task arrival (or, strictly, allocation by agents in the system) can significantly affect the quality of the global solution, and hence the number of tasks that are satisfied. The aim of concurrent negotiation is to alleviate the impact that task arrival has on solution quality. Specifically, it is of benefit if:

1. there are selfish coordinating agents which are not willing to cooperate without reward, and
2. a sub-task t_j of task M cannot find an available sensor of a particular type γ_i, t_j fails and, consequently, M fails. In many cases, t_j cannot satisfy its sensing requirement δ_j for sensor type γ_i not because there is no such sensor within t_j's range, but because there are sensors of type γ_i within its range that are allocated to other sub-tasks. If one such sub-tasks can find a replacement, that sensor can be allocated to t_j and, thus, t_j will succeed.

The negotiation mechanism detailed in this section allows an agent (buyer) representing a task to negotiate concurrently with other task-agents (sellers) to obtain a resource of type γ that is currently allocated to one of these other tasks in exchange for a fraction of its profit. Obviously, the buyer will only be interested in instances of resource type γ that it can make use of (i.e. utility of the resource

instance to the buyer is not 0). The buyer will negotiate simultaneously with all the sellers that currently employ a resource of type γ.

The buyer and the sellers work to different negotiation deadlines, each representing availability in terms of both resource and processing power. They follow a Sequential Alternating Protocol where at each step an agent can either accept the offer from the opponent, propose a counter-offer, renege from its commitment or opt out of the negotiation (typically if its deadline is reached). At each negotiation time period, the interest of each agent is represented by a proposal ϕ, which refers to the profit that will be paid to the seller by the buyer.

The buyer agent (B) consists of two main components: a *coordinator* and a number of *negotiation threads*. The negotiation threads deal directly with the sellers (one per seller S) and are responsible for deciding what counter-offers to send and what proposals to accept. Each thread inherits the preferences from the main buyer agent, including the acceptable ranges of values for the profit, the deadline of the negotiation and the current reservation value (the highest profit value that the buyer is willing to pay). The coordinator decides the negotiation strategies for each thread. If a thread reaches a deal with a particular seller, it terminates and notifies the coordinator. The coordinator will then notify all other negotiation threads of the new reservation value.

In this way, the buyer, B, will engage in simultaneous negotiations with all the sellers that currently possess a resource of type b. In our model, the buyer can either choose to terminate all negotiation threads once an agreement has been reached (*simple negotiation mode*) or it can wait until all the negotiations have been finished and then select the agreement that is most valuable (*extended negotiation mode*) either with the smallest profit to pay or with the highest utility achieved.

For each seller, if the negotiation succeeds, it will have to give up one of its resources to the buyer. As a result, it is necessary for the seller to obtain a replacement resource before it can enter the negotiation. If there is an available and appropriate alternative resource (i.e. a resource that achieves the requirements of the task — validates the allocation — without the original resource), it can replace the previously allocated resource with the alternative. We label this situation as 1-sequence negotiation. However, there exists a more complicated case (2-sequence) in which the seller needs to negotiate with another seller for a replacement resource before it can negotiate with the buyer (i.e. buyer B and seller S are negotiating about a resource b but S needs to negotiate with seller C about resource b' which is the replacement for b). If the seller cannot manage to find the replacement resource, it will not enter the negotiation.

The agents bargain about the profit that will be paid for the resource that the seller is currently holding (the price being a share of the profit that the buyer acquires in completing its task). The buyer and sellers use different negotiation strategies that are based on the set of linear strategies as specified in [9]. This strategy family is employed because it represents the neutral stances of both the buyer and the seller, not favouring anyone in particular and allows a solution to be found that is beneficial for both parties rather than having only one better

off. Furthermore, by doing so, it will increase the chance for more agents to participate and in turn, improve the global goal of maximizing the number of successful task allocations.

Specifically, a strategy is a sequence of decisions that an individual agent will make during negotiation. These decisions could be either to send an initial offer to the opponent, select an offer to propose, accept the offer proposed by the opponent or withdraw from the negotiation. Here, the value of the profit is between the minimum and the maximum limit of each agent. For the buyer, the proposed profit will increase in value over time and conversely, the seller's value will decrease. For each seller, the reservation value or the minimum profit (min_{PS}) it will accept is the difference between the profit it received by having the resource s and that received with the replacement resource s'. For example, if s receives a profit of 1.5 with s and a profit of 1.2 with s', the minimum profit it will accept from B is 0.3. The maximum profit (max_{PS}) it can expect from the buyer is the difference between the profit with s' and the maximum profit it can obtain. This is the incentive for the seller agents to enter into negotiation. For a seller S, at any time t between 0 and its negotiation deadline $t_{S_{max}}$, the value of the proposal it will send to B is: $\phi(S \rightarrow B) = max_{PS} - (max_{PS} - min_{PS}) \times (\frac{t}{t_{S_{max}}})^{\frac{1.00}{\beta_S}}$ where β_S is the parameter that defines the shape of the function.

On the other hand, the buyer will attempt to give up as little of its profit as possible. Thus, its minimum profit (min_{PB}) it is willing to pay is 0. The reservation value (max_{PB}) it is willing to pay is set at half of potential profit it can obtain if s is allocated. If it is higher, the buyer might not get any profit at all and it might not be tempted to enter the bargaining process. Thus, at any time t between 0 and deadline $t_{B_{max}}$, the value of the proposal B will send to S is: $\phi(B \rightarrow S) = min_{PB} + (max_{PB} - min_{PB}) \times (\frac{t}{t_{B_{max}}})^{\frac{1.00}{\beta_B}}$ where β_B is the parameter that defines the shape of the function.

When an offer proposed by a party is between the minimum and the maximum acceptable profit of the other party, it will be accepted and a provisional agreement (or deal) is created. If the negotiation is in the simple mode, the buyer will terminate all other negotiation threads and select the resource in the deal reached with the winning seller. If, however, it is in the extended mode, the buyer will attempt to establish as many deals as possible, and then commit to the best (based on its selection criteria), declining all others. The selection criteria that the buyer has in this model are (i) the deal with the least amount of profit, and (ii) the deal that can provide the highest utility value. The final agreement and the final allocated resource plays an important role in determining the success rate of subsequent tasks and this is reflected in the results of our empirical evaluation presented in Section 5.

5 Evaluation

Having defined our negotiation mechanism, we now present a detailed discussion of our empirical evaluation aimed at assessing the benefit of employing our concurrent negotiation mechanism in sensor-task allocation.

The sensors and tasks are deployed in uniformly random locations in a 400m $\times$ 400m environment. Each sensor range (r_i) is randomized between 20m and 40m and their maximum utility is calculated as $(r_i/40)^2$, which ensures that their the values lie between 0.25 and 1. The operational ranges of the sub-tasks are set to be randomized between 40m and 80m. The values for β_B and β_S are selected randomly between 0.95 and 1.05. The threshold ξ is set at 0.75.

The task arrival rates are controlled by the *task_per_hour* parameter, which ranges from 2 to 8, and *number_of_days* parameter, which is kept at 2 days. Each task can last for an arbitrary amount of time, ranging from 5 minutes to 4 hours. There are *total_sensor_types* different sensor types, which will vary between 4 and 8 and, for each sensor type, there will be *total_sensors_per_type* sensors. For each task, the number of sub-tasks will be varied between 4 and 5. Each sub-task type will require a number of different sensor types, which varies between 1 and 4. These individual sensor type requirements are generated randomly and have the value between 1 and *total_sensor_types*.

To evaluate the negotiation mechanism, we benchmark our model with 3 different settings: *4tph 4st*, *4tph 8st* and *8tph 8st* where *tph* stands for *task_per_hour* and *st* stands for *total_sensor_types*. With each setting, we vary *total_sensors_per_type* between 30 and 250 to create additional 12 environments, each then carries further 500 experiments with randomized data sets. The results are averaged and put through a regression test to ensure that all differences are significant at the 99% confidence level.

We measure the number of successful tasks, the average profit achieved and the running time. We also measure the performance of the different negotiation modes: simple mode (terminate whenever an agreement is reached); and extended mode with either smallest profit or highest utility selection criterion. It would be reasonable to expect that the different ratios between the number of tasks and sensors leads to different improvements in the number of successful tasks between negotiation-enabled and non-negotiation models. For example, when the number of tasks remain unchanged, the more sensors there are, fewer negotiations are required and thus, any improvement due to negotiation might decrease. Hence, we explored variations in these values.

We now turn to the specific results.

Hypothesis 1 *By negotiating, agents will have a better chance of finding a successful allocation as well as increasing the total profit achieved. Moreover, the running time of the algorithm is still acceptable*

To evaluate this hypothesis, we measure the number of successful allocated tasks and the total amount of profit achieved for the model with the 1-sequence negotiation featured in extended utility mode and the one without the negotiation feature. The differences are shown in Figure 2.

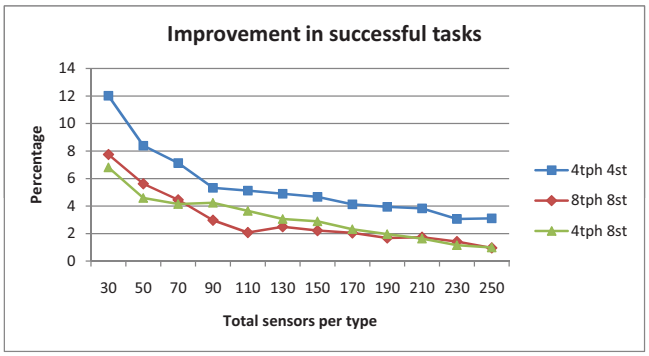

Fig. 2. The improvement of successful tasks between 1-sequence concurrent negotiation (simple negotiation mode) vs no negotiation.

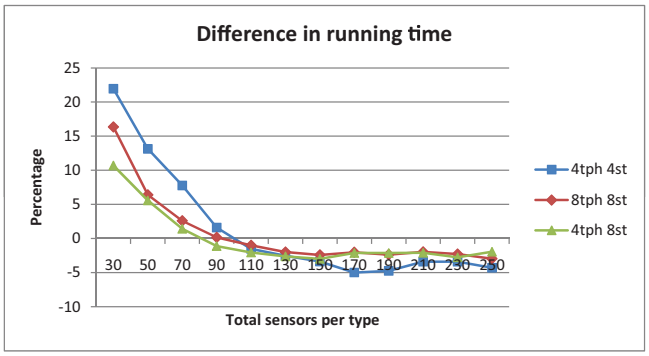

Fig. 3. The differences of the running time of the algorithm between 1 sequence concurrent negotiation (simple negotiation mode) vs no negotiation.

As can be seen, negotiation allows the number of successful tasks to increase in all cases, varying between 2% and 12%. This can be explained by the fact that, in many situations, a sub-task in the standard model fails because it cannot find a sensor of a particular type to satisfy its requirement. This same sub-task in the negotiation-enabled model can now bargain with another sub-task to acquire a sensor that is unavailable otherwise and this helps it to obtain a successful allocation and, eventually in some cases, lead to a successfully allocated task. As the number of successful tasks increases, the overall profit achieved also increases.

We detail the differences between the running time of our model with and without negotiation in Figure 3. This is the actual amount of time that the machine took to solve the allocation problem. As can be seen from the graph, the negotiation-enabled model takes longer than its counterpart when the number of sensors is roughly between 5% and 22% which, we believe, is still acceptable given the more beneficial outcomes achieved. However, as the number of sensors increases, the time it took decreases such that there is a negligible impact on

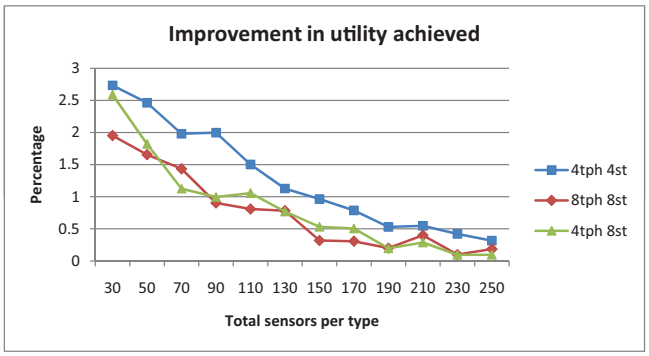

Fig. 4. The improvement of utility achieved between 1 sequence concurrent negotiation (simple negotiation mode) vs no negotiation.

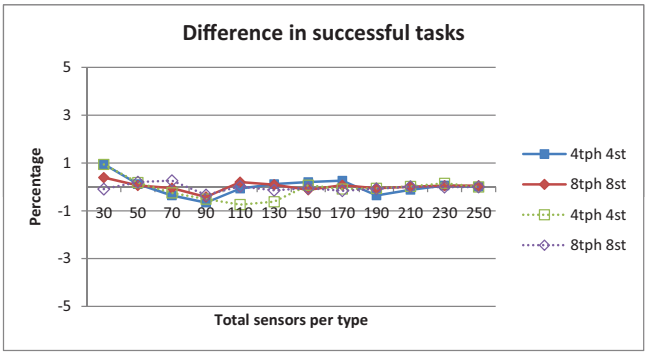

Fig. 5. The differences of the number of successful tasks between 1-sequence highest utility agreement (straight line) vs 1-sequence lowest profit (dotted line) vs simple negotiation mode.

running time. By far, the greatest impact on running time is the number of tasks and sensors involved in a problem.

Hypothesis 2 *The overall utility achieved through the use of negotiation is higher than that without.*

The differences between the averaged utility achieved by using model with and without negotiation feature are displayed in Figure 4. As can be seen from hypothesis 1, negotiation enabled model allows higher number of successfully allocated tasks in all situations. Consequently, the utility achieved by successful tasks is increased, leading to an increase in the averaged utility obtained by a task. Also similar to hypothesis 1, the more sensors there are, the lower this increase will be.

Hypothesis 3 *There is no clear advantage of selecting the extended negotiation mode.*

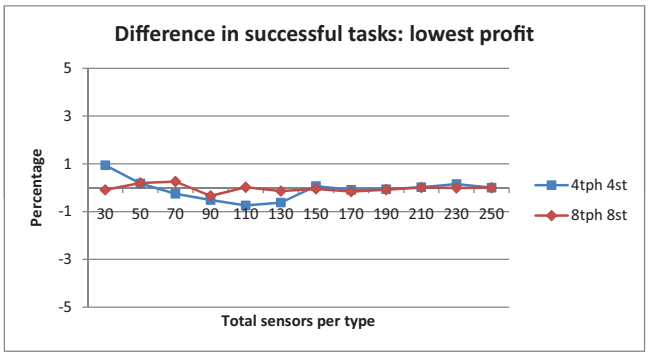

Fig. 6. The differences of the number of successful tasks between 1-sequence lowest profit agreement vs simple negotiation mode.

To evaluate this hypothesis, we show the difference between the performance of 1-sequence lowest profit agreement and 1-sequence highest utility agreement vs simple negotiation mode in Figure 5.

As can be seen, the difference between extended negotiation mode and the simple negotiation mode are negligible with the highest value less than 1%. There is no decisive pattern of which negotiation mode provides a more desirable outcome. Obviously, the extended negotiation mode strongly favours the buyer sub-task (see Section 4) whereas the simple negotiation mode treats all agents equally. Consequently, it is rational to select the simple mode as the negotiation method since the sellers will be more willing to participate (they do not have to wait for the buyer to finalize their agreements). Moreover, it will be faster for an agreement to be reached.

Hypothesis 4 *Allowing 2-sequence negotiation in the model provides higher number of successful allocated tasks than 1-sequence negotiation enabled model.*

2-sequence negotiation allows a sub-task agent to have a slightly better chance of finding a replacement sensor (see Section 4). For most sellers, instead of only finding free sensors, they can now negotiate with other potential seller for a replacement sensor, having both the roles of buyer and seller at the same time. By doing so, the chance of finding a replacement sensor for any seller is increased and that results in a higher number of negotiations for the original buyer and, consequently, a higher number of successful negotiations, eventually leading to an increase in the number of successful negotiations compared to its 1-sequence counterpart. The results are clearly demonstrated in Figure 7.

Hypothesis 5 *The running time of 2-sequence negotiation enabled model is considerably longer than that of 1-sequence counterpart.*

Even though 2-sequence negotiation mode provides better outcomes than 1-sequence mode, the running time of the algorithm is much higher (see Figure 8).

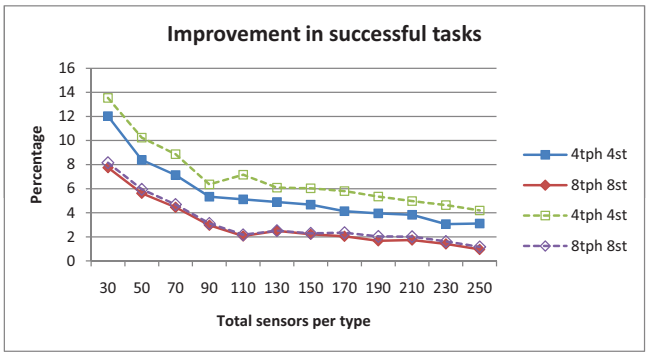

Fig. 7. The differences of the number of successful tasks between 2-sequence *vs* 1-sequence concurrent negotiation (dotted *vs* straight lines).

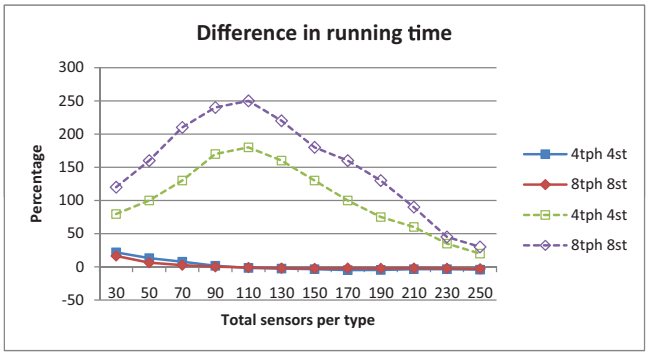

Fig. 8. The differences of the algorithm running time between 2-sequence *vs* 1-sequence concurrent negotiation (dotted *vs* straight lines).

In the worst case, it is nearly 2.5 times worse and even in the best case, it takes nearly 50% longer than its counterpart.

Now that the sellers can negotiate with other potential sellers, their chances of finding a replacement is increased but also the number of negotiations carried out is also increased. There is no way of knowing which negotiation will be beneficial and thus, all the negotiations will need to be carried out. As a result, there will be many unnecessary bargaining processes, leading to a dramatic increase in the running time of our model.

As can be seen, even though the number of successful tasks increases with 2-sequence negotiation, the time it takes to complete is considerably longer than that of 1-sequence counterpart. Thus, it will not be beneficial to support more than 2-sequence negotiation in our model since the trade-off between the successful task and the running time will be undesirable.

6 Discussion and Related Work

There are only a small number of sensor-task allocation studies that have considered the heterogeneous sensor, heterogeneous task case [11, 5] and our work falls in this class, which can be considered the most generic version of the sensor-task allocation problem. In addition, the problem we are considering can be viewed as a more general problem of resource allocation such as scheduling jobs on unrelated parallel machines [16] (the feasible constraint is that a job may need to be performed by a set of families of machines) or the Bin Covering problem (our problem is a generalization of this problem when the item may take a different amount of space in different bins). Our MRA algorithm presented in Section 3 is an adaptation of the MRGAP algorithm proposed in [5] in which the idea is to consider tasks as knapsacks that together form an instance of the Generalized Assignment Problem (GAP).

Resource allocation models in multi agent systems have two major branches: centralised and decentralised [1, 4]. Centralised systems make use of a single agent to assign resources to all tasks and optimal outcomes might be achieved because that single agent has a global view of the situation. The most successful centralised models are auctions and it comes in various form including regular or combinatorial auctions [6]. Agents may submit the "best" bid(s) serving their own interests and wait for the final allocation decided by the auctioneer. In addition, advantage of such models is that the communication protocols required are normally simpler than that of decentralised approaches [1]. Nonetheless, the central agent creates a bottleneck and generally, these solutions do not scale well. Decentralised systems are typically preferred in practical situations [4] and peer-to-peer negotiation has long been a popular technique for agent coordination in such system.

In sensor networks, various forms of negotiation have been explored. For example, Sujit *et al.* [15] employ an auction-based negotiation model for distributing UAVs (Unmanned Aerial Vehicles) to search and attack some targets in the environment. Similarly, Shima *et al.* [14] use an auction-based negotiation model to establish information regarding other neighbouring nodes and estimate costs for other members to assign to different targets in order to find an efficient solution for all the participating nodes. The DISTINCT algorithm [12] uses negotiation to distribute tasks among robots. The disadvantage of these approaches is that they cannot guarantee all the negotiations will terminate after a finite number of cycles.

Another model introduced by Howard *et al.* [3] uses a market-based approach and the contract net protocol to allocate a group of robots to a number of tasks. Each task is announced and all the robots bid for tasks. If a robot has already been allocated to another task then the robot will select the better task and broadcast the other. The major issue with this model is that there are a great deal of duplicate allocations, resulting unnecessary time and resource consumed.

In [2, 7], Kulik *et al* introduce four SPIN (Sensor Protocols for Information via Negotiation) protocols for exchanging information in wireless sensor networks. They are all negotiation based and can be applied in either point-to-point or

broadcast modes. In either mode, the sensor nodes use some variation of the three-stage handshake protocol to negotiate for newly discovered data. Basically, whenever a sensor discovers new data, it will broadcast its findings (ADV message) to its neighbouring sensors. These sensors, in turn, will decide whether or not to ask for the actual data to be sent to them (REQ message) based on their constraints. Finally the initiator will response to the REQ message with a DATA message containing the actual data. Even though the communication between sensors can be reduced by using these protocols, the sensors need to be equipped with large buffers to store previous requests/data to avoid duplication. Moreover, these protocols only provide best results when the topology of the network is fixed.

As can be seen, using negotiation as the sole means to allocate resources might not be beneficial. However, it is useful if negotiation is used to enhance existing allocation algorithms. There are a number of negotiation models that can be employed such as auctions, double auctions or bilateral negotiations. However in this work, we consider the application of the multiple concurrent bilateral negotiation model introduced by Nguyen *et al.* [10] since it allows the agents to engage in real time and the results obtained are close to optimum [10, 9]. There are a number of shortcomings with our model, however. First, the strategies employed by the agents are linear and constant throughout each encounter. Ideally, they should adapt to their opponents so that the participating agents might be able to obtain better outcomes. Second, we consider profit to be exchangeable between tasks so that it can be used as the base for the negotiations to happen. This is not always an appropriate assumption and this issue requires further investigation.

7 Conclusion

In this paper, we have proposed a decentralised agent-based approach for handling the sensor-task allocation problem in dynamic environments where the tasks and resources can appear/disappear any time. Moreover, our model allows various tasks to compete for the same resources in a graceful manner. In particular, we have incorporated a negotiation mechanism as a post-processing stage of agent-based allocation models. The mechanism allows resources to be exchanged between self-interested agents. Specifically, a task negotiates concurrently with other tasks to obtain a resource that is currently allocated to one of these tasks in exchange for a fraction of its profit which it will receive if it can obtain a valid alternative allocation. Via empirical evaluation, we have demonstrated that this offers significantly better results when compared with an agent-based allocation model without resource re-allocation.

References

1. Chevaleyre, Y., Dunne, P.E., Endriss, U., Lang, J., Lemaitre, M., Maudet, N., Padget, J., Phelps, S., Rodrguez-aguilar, J.A., Sousa, P.: Issues in multiagent resource allocation. Informatica 30, 2006 (2006)

2. Heinzelman, W.R., Kulik, J., Balakrishnan, H.: Adaptive protocols for information dissemination in wireless sensor networks. In: Proceedings of the ACM MobiCom 99. pp. 174–185. Seattle, Washington (1999)
3. Howard, A., Viguria, A.: Controlled reconfiguration of robotic mobile sensor networks using distributed allocation formalisms. In: Proc. of the NASA Science Technology Conference (NSTC 2007) (2007)
4. Jacyno, M., Bullock, S., Payne, T., Luck, M.: Understanding decentralised control of resource allocation in a minimal multi-agent system. In: AAMAS 2007: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems. pp. 1–3 (2007)
5. Johnson, M.P., Rowaihy, H., Pizzocaro, D., Bar-Noy, A., Chalmers, S., Porta, T.F.L., Preece, A.: Frugal sensor assignment. In: Nikoletseas, S.E., Chlebus, B.S., Johnson, D.B., Krishnamachari, B. (eds.) DCOSS. Lecture Notes in Computer Science, vol. 5067, pp. 219–236. Springer (2008)
6. Krishna, V.: Auction Theory. Academic Press (2002)
7. Kulik, J., Heinzelman, W.: Negotiation-based protocols for disseminating information in wireless sensor networks. Wireless Networks 8, 169–185 (2002)
8. Le, T.P., Norman, T.J., Vasconcelos, W.: Agent-based sensor-mission assignment for tasks sharing assets. In: Proceeding of the Third International Workshop on Agent Technology for Sensor Networks. Budapest, Hungary (May 2009)
9. Nguyen, T.D.: A heuristic model for concurrent bilateral negotiations in incomplete information settings. Ph.D. thesis, University of Southampton, Southampton, England (2005)
10. Nguyen, T.D., Jennings, N.R.: Coordinating multiple concurrent negotiations. In: Proceedings of the third International Joint Conference on Autonomous Agents and Multi Agent Systems. pp. 1064–1071. New York, USA (2004)
11. Preece, A., Pizzocaro, D., Borowiecki, K., de Mel, G., Gomez, M., Vasconcelos, M., Bar-Noy, A., Johnson, M.P., La Porta, T.L., Rowaihy, H., Pearson, G., Pham, T.: Reasoning and resource allocation for sensor-mission assignment in a coalition context. In: MILCOM 2008 (2008)
12. Salemi, B., Will, P., min Shen, W.: Distributed task negotiation in modular robots. In: Robotics Society of Japan, Special Issue (2003)
13. Sensoy, M., Le, T., Vasconcelos, W.W., Norman, T.J., Preece, A.D.: Resource determination and allocation in sensor networks: A hybrid approach. Computer Journal p. to appear (2010)
14. Shima, T., Rasmussen, S., Chandler, P.: UAV team decision and control using efficient collaborative estimation. Proceedings of the 2005 American Control Conference 6, 4107–4112 (2005)
15. Sujit, P.B., Sinha, A., Ghose, D.: Multiple UAV task allocation using negotiation. In: AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems. pp. 471–478 (2006)
16. Sung, S.C., Vlach, M.: Maximizing weighted number of just-in-time jobs on unrelated parallel machines. Journal of Scheduling 8(5), 453–460 (2005)