

# Assigning Sensors to Competing Missions

Hosam Rowaihy\*, Matthew Johnson<sup>§</sup>, Amotz Bar-Noy<sup>§</sup>, Theodore Brown<sup>§</sup> and Thomas La Porta\*

\*Department of Computer Science and Engineering  
Pennsylvania State University

<sup>§</sup>Department of Computer Science  
City University of New York

**Abstract**—When a sensor network is deployed in the field, it is typically required to support multiple simultaneous missions, which may start and finish at different times. Schemes that match sensor resources to mission demands thus become necessary. In this paper, we propose centralized and distributed schemes to assign sensors to missions. We also adapt our distributed scheme to make it energy-aware to extend network lifetime. Finally, we show simulation results comparing these solutions. We find that our greedy algorithm frequently performs near-optimally and that the distributed schemes usually perform nearly as well.

## I. INTRODUCTION

In many sensor network applications, it is necessary to support multiple missions that may arrive over time and compete for the same sensing resources. For *isotropic* sensing devices that record ambient temperature, for example, the information provided by a single sensor can be used to support multiple missions given that they lie within its sensing range. A directional or *anisotropic* sensor, such as camera, however, must be directed to a certain location and hence can in general only support a single mission. Thus, *choices must be made* as to which sensors will be assigned to which missions. Given information about available sensors and missions, the network must have a process to choose the “best” assignment of sensors to missions. An intelligent sensor network should direct its resources to the most important feasible missions, reallocating resources appropriately as missions arrive or depart, while taking care not to waste resources on unsuccessful missions.

In some networks, there may be a static set of long-term missions, in which case the aspect of time may be eliminated. In other settings, mission arrivals and departures may be infrequent, so that for each block of time, sensor assignment can be solved as a static problem. Even in this static setting, our problem is computationally hard to solve optimally. Thus we use approximation algorithms and heuristics.

A centralized approach for sensor assignment will collect all the relevant information at a central location for decision-making and then distribute assignments. Such an approach can be expensive in terms of communication overhead. Another approach is to let nodes\* make these assignment decisions locally, in a distributed manner, using mission information that is disseminated into the network. While this should decrease communication costs, a centralized algorithm may be able to guarantee better solution.

In this paper, we consider the problem of assigning directional sensors to missions in wireless sensor networks. We consider both centralized and distributed approaches to solving

the dynamic problem. The distributed method we propose is a novel multi-round proposal scheme, which we adapt to work in the dynamic setting. We also provide an energy-aware extension to the distributed scheme to extend network lifetime.

Our simulations show that in spite of the provable worst-case difficulty, efficient schemes can perform near-optimally. Furthermore, distributed schemes are often competitive with the centralized greedy algorithm, especially in networks with high node density.

## II. RELATED WORK

There has been some work in defining frameworks for sensor-mission assignment problems. For example, [2] defines a framework for the assignment problem in which the goal to maximize the utility while staying under a predefined budget. However, the authors do not consider the case of competing missions. The general problem of sensor selection to achieve an objective has also received sizable attention lately. For example, in [5], [7] the authors solve the coverage problem, which is a related problem, using the least number of sensors to conserve energy. Another related problem is to efficiently locate and track targets such as the works in [8] and [3]. The problem we consider here is different from previous work since it considers multiple missions with different priorities. These missions contend for the same set of sensors which calls for resolution mechanisms.

The Semi-Matching with Demands (SMD) problem for sensor-mission assignment was recently introduced in [1]. The original SMD problem is less general than the problem we consider here in two ways. First, it uses a stricter profit model with no credit for partial mission satisfaction. Second, it is defined only for a static set of missions, rather than for a dynamic environment, which is our focus here.

## III. NETWORK AND PROBLEM MODELS

We begin this section by discussing our network model and then formally defining our sensor-mission assignment problem which is a generalization of the SMD problem [1]. We then extend our problem model from the static setting to a dynamic setting in which missions appear and disappear over time.

### A. Network Model

In our network model, we assume a set of static sensors pre-deployed in a field. Missions can arrive and depart over time. By a mission, we mean a primitive sensing task that requires information, which may be contributed by one or more deployed sensors. Each mission is defined by a specific geographic location. An example of a mission is video

\*In this paper we use the terms *sensor* and *node* interchangeably.

monitoring an area of interest. General missions that cover large areas, such as perimeter monitoring, can be divided into multiple missions each having its own location. The deployed sensors are directional in nature and hence each of them can be assigned to a single mission (i.e. directed to one location).

### B. Problem Model

The problem instance is modeled as a weighted bipartite graph, whose vertices consist of a set of sensors  $\{S_i\}_i$  and a set of missions  $\{M_j\}_j$ . A positively weighted edge  $(S_i, M_j)$  means that sensor  $S_i$  is applicable to mission  $M_j$ . The weight of the edge  $(e_{ij})$  indicates the utility or quality of information that  $S_i$  will contribute to  $M_j$  if this assignment is chosen. Each mission  $M_j$  is associated with a positive demand value  $d_j$  and a positive profit value  $p_j$ . The demand indicates the total utility the mission desires, which may be contributed by one or more sensors. Profit for mission  $M_j$  indicates the importance of the mission and is awarded based on the percentage of satisfied demand, but only if this percentage reaches a satisfaction threshold  $T$ .  $p_j$  is the maximum profit receivable for mission  $M_j$ . Note that there need not be any relation between a mission's demand and profit. To simplify the problem we assume that the utility amounts received by a mission are additive. That is the total utility received by a mission is equal to the sum of the utilities provided by sensors assigned to it. While this may be realistic in some settings, in others it is not; for our purpose of comparing the different algorithms this assumption is sufficient.

What is sought is a *semi-matching* of sensors to missions, so that (ideally) each mission's demand is fully satisfied. That is, a sensor may be assigned to at most one of the missions to which it is applicable, but a mission may accept utility from multiple sensors. Of course, fully satisfying all missions may not be feasible; the goal in general is to maximize total profits. We call this problem *Threshold SMD*. The problem instance and goal are formally defined as follows:

**Instance:** A global threshold  $T \in [0, 1]$  and a weighted bipartite graph  $G = (S, M, P, D, E)$ , where  $S = \{S_1, \dots, S_n\}$  is a collection of sensors and  $M = \{M_1, \dots, M_m\}$  is a collection of missions; each mission  $M_j$  is associated with a profit  $\{p_j\}$  and a demand  $\{d_j\}$ ; each edge in  $S \times M$  has an edge-weight  $e_{ij}$  indicating utility.

**Goal:** Find a semi-matching  $F \subseteq S \times M$  (no two chosen edges share the same *sensor*), in which  $\sum_j p_j(u_j)$  is maximized, where  $u_j$  is the total utility received by mission  $M_j$  divided by demand  $d_j$ . The profit functions are defined as follows:

$$p_j(u_j) = \begin{cases} p_j, & \text{if } u_j \geq 1 \\ p_j \cdot u_j, & \text{if } T \leq u_j < 1 \\ 0, & \text{if } u_j < T \end{cases}$$

This problem may be formulated as a mathematical program. The program below employs two sets of decision variables:  $x_{ij}$  indicating whether sensor  $S_i$  is assigned to mission  $M_j$ , and  $u_j$  indicating mission  $M_j$ 's satisfaction level ( $u_j = \sum_{i=1}^n x_{ij}e_{ij}/d_j$ ).

$$\begin{aligned} \text{Maximize:} & \sum_j p_j(u_j) \\ \text{Such that:} & \sum_{i=1}^n x_{ij}e_{ij} \geq d_j u_j, \text{ for each mission } M_j, \\ & \sum_{j=1}^m x_{ij} \leq 1, \text{ for each sensor } S_i, \\ & x_{ij} \in \{0, 1\} \forall x_{ij} \text{ and } u_j \in [0, 1] \forall u_j \end{aligned}$$

The threshold-based problem generalizes both all-or-nothing profits (with  $T=1$ ) and fully fractional profits (with  $T=0$ ). When  $T=1$ , profit  $p_j$  is received only for fully satisfying mission  $M_j$ . In this case, the program reduces to an Integer Program (IP) with mission satisfaction level  $u_j \in \{0, 1\}$ , which is the strict SMD problem of [1].

The corresponding Linear Program (LP), in which all variable constraints are relaxed from  $\{0, 1\}$  to  $[0, 1]$  and  $T$  is relaxed to 0, allows for fractional profits to be awarded for partial satisfaction and for sensors to be fractionally assigned to multiple missions. This fully fractional version can be solved optimally by LP. Such an *optimal fractional* solution (see Section VI) provides an upper bound on the true optimal solution value. With a non-zero threshold value, however, the objective function is neither continuous nor concave, and so standard LP techniques do not apply.

Treating  $T$  as part of the problem instance therefore means that this formulation can only be harder than the original strict version (which was shown to be NP-Complete in [1]). Intuitively, lowering the threshold should make the problem easier. However, we prove in [6] that the problem is NP-Complete even with threshold 0.

### C. Dynamic Problem Model

We now provide an orthogonal generalization of the original problem in terms of time. The problem statement is the same, except that now each mission is associated with a start time and an end time. A mission's demand and maximum profit are constant over time. Awarded profit for a mission is computed at each discrete timestep, based on the satisfaction level at that instant. Total profit for a mission is simply the sum of the instantaneous profits. We do not require that a mission's demand be met over its entire lifetime in order to receive profit. So, our profit model is in this sense fractional *in terms of time*. The dynamic version is thus given by essentially the same mathematical program given above except that each variable now has an additional time index.

As a generalization of the static problem, previous hardness results apply to the dynamic version. Indeed, a natural strategy for this version is to solve the static problem at each timestep.

## IV. CENTRALIZED SOLUTION

In this section we present a *greedy* centralized algorithms to solve the sensor-mission assignment problem. In such a scheme all the matching decisions are made in the base station (which is a special control node in the network) that has complete knowledge of all missions and sensors. To do this, the base station needs first to inform all nodes in the network about the missions, their demands and profits. Then, each sensor that is within sensing range of at least one mission will send back a message with a list of missions it can support and its potential contribution value to each of them.

After that, the base station runs the assignment algorithm and sends assignment instructions to the appropriate sensors. The dynamic problem is treated as a series of static problem instances in which case the algorithm will find a sensor assignment every time a mission arrives or departs.

The greedy algorithm repeatedly satisfies the most *currently profitable* mission, i.e. the mission that can be satisfied with the greatest profit, using currently available sensors. If  $S' \subset S$  is the set of not-yet-assigned sensors (initially  $S' = S$ ) and  $u_j = \sum_{S_i \in S'} e_{ij}/d_j$ , then the profit currently achievable by mission  $M_j$  is  $p_j(u_j)$ . Of course, it may be that not all sensors are needed to achieve this profit; conversely, if the demand threshold is not met, this profit is 0. The algorithm repeatedly selects mission  $M_j$  with maximum current profitability, and satisfies it with available sensors in order of decreasing utility value  $e_{ij}$ , until either  $M_j$  is *fully* satisfied or all sensors with non-zero offers to  $M_j$  have been used. Used sensors are then removed from  $S'$ . When there are no remaining missions with non-zero current profitability, the algorithm completes.

When  $T = 1$ , this algorithm produces the same result as the simpler greedy algorithm of [1]. We prove (in [6]) that our greedy algorithm has the same  $\Delta$ -approximation performance guarantee.  $\Delta$  is the maximum mission degree, which is the maximum number of sensors that make a non-zero offer to any mission.

For geographic settings with limited sensing range, we also extend (in [6]) the approximation scheme (PTAS) of [1] to the threshold problem. Although it is theoretically efficient, we found in our experiments that achieving performance competitive with the greedy algorithm's requires an unreasonable computation time, and so we omit the results here.

## V. DISTRIBUTED SOLUTIONS

Although centralized schemes may provide better solutions to the matching problem due to their global view of the field, they can have high communication cost. Because a centralized scheme requires information about all nodes in the network such as their locations and utilities to the different missions, the number of messages required to be sent to the base station can be very large, especially in dense networks. This communication cost becomes even higher once we consider a dynamic system in which missions arrive and depart at different points in time, requiring the base station to continually gather information about nodes.

To avoid this cost, we consider distributed schemes. In such an approach, a *mission leader* is selected for each mission. This should be a node that is close to the mission's location (geographic-based routing techniques such as [4] can be used to do this). The leaders are informed about the missions' demands and profits by the base station. Then they run a local process to match nearby sensors to the missions. Since the sensors have limited sensing range, only nearby nodes are considered. In this section, we consider a multi-round proposal scheme for the static setting, which we then adapt to dynamic cases. We also propose an energy-aware extension which helps in prolonging network lifetime.

### A. Multi-Round Proposal Scheme (MRPS)

In this scheme, each mission leader advertises its mission information (location, demand and profit) to nearby nodes (e.g. neighbors within two hops). When a nearby sensor hears such an advertisement message for one or more missions, it sends a single proposal to the mission it perceives to be the best match. The ranking of missions is based on the mission profit and the fraction of the mission demand that the sensor can satisfy. Using the notation of Section III, sensor  $S_i$  ranks mission  $M_j$  according to  $B_{ij} = e_{ij}p_j/d_j$ . The leader, on the other hand, selects proposing sensors greedily based on their contribution. If the leader does not select a proposing sensor, in the next round the sensor proposes to the next mission on its list. The scheme consists of a series of proposal-reply rounds. The more rounds allowed, the better the matching may be. However, as the number of rounds increases, the communication cost grows and we may obtain diminishing returns.

Since our aim is to achieve the highest profit from *successful* missions, we use a mechanism to prevent missions that will never be fully successful from holding up sensors that could help other missions. In each round, mission leaders assess the satisfaction level of their missions. If it is not greater than an increasing threshold value ( $\alpha(k)$  for round  $k$ ) then the mission is deemed unattainable and the sensors are released. This threshold is initialized to a fixed value and incremented each round. After a sufficient number of rounds, it will reach the global threshold  $T$ , the preset threshold value for success, at which time all missions that are not yet successful release their sensors. The rising threshold yields two benefits: (1) it increases the chance that the most satisfied missions will become fully satisfied and (2) it prevents sensors from wasting their energy on unsuccessful missions receiving no profit.

### B. Dynamic Proposal Scheme (DPS)

MRPS is designed for static cases in which many competing missions are dealt with simultaneously. By handling missions as they arrive, in dynamic settings, however, we may expect to have less competition for sensing resources. So, we opt for a lighter-weight scheme that does not require multiple rounds. We call this scheme the *Dynamic Proposal Scheme* (DPS).

As in MRPS, each mission has a leader which advertises mission information to nearby nodes, when the mission arrives. A sensor that hears this announcement can be in one of two states: (1) *not assigned*, in which case it proposes to the mission with its utility or (2) *assigned to a mission*, in which case the sensor calculates its effective profit for both missions (the  $B_{ij}$  value above) and chooses either to stay with the current mission or to propose to the new mission, depending on which value is higher.

After the mission leader collects the proposals, it tries first to satisfy the mission with sensors in the *not assigned* state by greedily picking sensors with highest utility. If these do not suffice, it may *preempt* other missions by *stealing* sensors in the *assigned* state from other ongoing missions. If satisfaction threshold  $T$  is achieved, then the mission leader sends assignment messages to the respective sensors which

start collecting information to support the mission. Otherwise, no assignments are made. If a selected sensor preempts an existing mission, the procedure below is followed.

Suppose a new mission,  $M_j$  with leader  $L_j$ , started in an area close to an ongoing mission,  $M_k$  with leader  $L_k$ . If a sensor  $S_i$  currently assigned to some  $M_k$  decides that it will generate greater profit by contributing to  $M_j$ , it notifies  $L_k$  of its intention.  $L_k$  then tries to find one or more sensors to replace  $S_i$ . If no such node(s) are found, the leader will agree on the reassignment as long as its current satisfaction level does not drop below  $T$ , which would cause the mission to fail. In this case, the reassignment is temporarily denied. If the new mission  $M_j$  will fail without  $S_i$ , then a second test is performed. If  $M_j$ 's current profit (with  $S_i$ ) is greater than  $M_k$ 's, the leader of  $M_k$  will release its hold on  $S_i$  and agree to the reassignment even if it will cause its own mission to fail. The reassignment becomes final once  $S_i$  is selected by  $M_j$ . Only at that time are the replacement sensor(s) activated. To reduce both the interruption of ongoing missions and the communication overhead, no cascading preemption is allowed. That is, if mission  $M_j$  preempts mission  $M_k$ ,  $M_k$  will try to satisfy its demand only with available sensors rather than by stealing sensors from a third mission.

When a mission ends, the leader sends out a message to announce that the mission has ended and all its assigned sensors are released. Because the system is dynamic, missions that have not reached their success threshold or are not fully satisfied after the first assignment process will retry to obtain more sensors if they think that there will be more available. This can happen when a nearby mission terminates and has its sensors released. This information can be obtained either from the base station or by overhearing the message announcing the end of a mission.

### C. Energy-aware Dynamic Proposal (EDPS)

A drawback of DPS is that it does not consider the remaining energy in sensors when making assignment decisions. It selects a sensor based only on the provided utility. As such, it is possible for one sensor's energy to be depleted while other, even nearby sensors retain high energy levels. By taking remaining energy levels into consideration when assigning sensors, we may extend the time during which live sensors remain in diverse locations in the field and thereby increase total profits. With this observation, we extend DPS to make it energy-aware and call it the *Energy-aware Dynamic Proposal Scheme* (EDPS).

Instead of using the sensor's provided utility alone to make the assignment decision, EDPS uses a function of utility  $U$  and fraction of remaining energy  $E$ . We define  $f(U, E) = U \cdot E^\beta$ , where  $\beta$  is a design parameter that controls the influence of remaining energy. If  $\beta$  is zero, EDPS reduces to DPS.

In order to consume energy more evenly among nodes, after the initial assignment, possible sensor candidates for a mission periodically update the leader with their current energy levels. The leader then checks how evenly these sensors' energy is being consumed. At that time, the leader may choose to change

the assignments of nodes by reapplying the decision function  $f$ . These periodic updates increase the communication overhead, but as will be shown in Section VI, the increase is not very large. As may be expected, this scheme works better in a dense sensor network, in which there are typically many choices available to the mission leader.

## VI. PERFORMANCE EVALUATION

To evaluate our algorithms we built a simulator in Java and tested them using randomly generated problem instances. We perform two sets of experiments to compare the performance of the centralized greedy scheme and the distributed schemes. The first test is for the static setting, in which the all missions take place simultaneously. The second test is for the dynamic setting, in which missions arrive and depart over time. For the dynamic case, we also demonstrate how EDPS can be used to extend network lifetime.

### A. Assumptions

Each mission has a demand, an abstract value of the amount of sensing resources it requires, which is exponentially distributed with an average of 2 and a maximum of 6. Also associated with each mission is a profit value, which measures its importance. The profit is also exponentially distributed, but with an average of 1. This simulates common scenarios in which many missions demand few sensing resources and a smaller number demand more resources. The same applies to profit. The profit obtained from a successful mission  $M_j$  is equal to  $p_j(u_j)$  as defined in Section III. We consider a mission successful if it receives at least 50% of its demanded utility from allocated sensors (i.e.  $T = 0.5$ ). Each sensor can only be assigned to a single mission.

The utility that sensor  $S_i$  provides to mission  $M_j$  is defined as a function of the distance  $D_{ij}$  between them. Many types of sensors exhibit some kind of quality deterioration or signal attenuation based on distance. In our experiments, we choose a simple function of distance as a representative example. In order to evaluate their utilities to missions, we assume that all sensors know their geographical locations. Formally, the potential utility contribution is:

$$e_{ij} = \begin{cases} \frac{1}{1+D_{ij}^2/c}, & \text{if } D_{ij} \leq SR \\ 0, & \text{otherwise} \end{cases}$$

where  $SR = 30m$  is the global sensing range. This models a typical signal attenuation which depends on the distance squared. We set  $c = 60$  to dampen this effect.

Nodes are deployed in uniformly random locations in a  $400m \times 400m$  field (the base station is located in the center of the left edge). Missions also are created in uniformly random locations in the field. The communication range of nodes is set to  $40m$ . When nodes are deployed we ensure that the network is connected. If a randomly created instance is not connected, it is discarded by the simulator.

Finally, we assume perfect communication channels with no errors or collisions. Hence, each message is sent only once and its delivery is guaranteed. All messages have the same

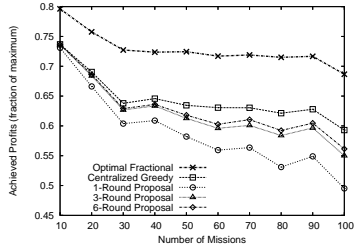


Fig. 1. Static: achieved profits

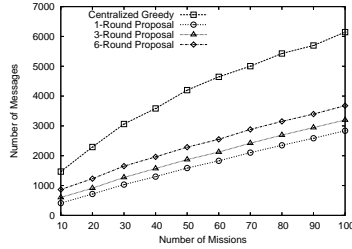


Fig. 2. Static: number messages

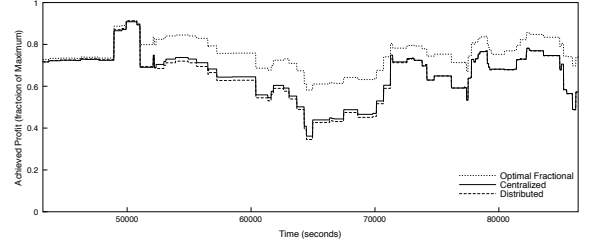


Fig. 3. Trace of network performance ( $\lambda = 6$  missions/hr)

size and hence only the number is considered when studying the communication overhead. For messages that travel over multiple hops, each hop counts as a single message in the total. When a broadcast message (e.g. mission advertisements by mission leaders or base station) is sent, it is received by all one-hop neighboring nodes and so we count it as one. All routing decisions are made based on a pre-configured routing table that follows the shortest path to destination.

### B. Static Scenarios

**Setup:** In this experiment all missions occur simultaneously, with the same start and end times. We fix the number of nodes in the field to 500 and vary the number of missions from 10 to 100. In the following results we show the average of 10 runs for achieved profits and number of exchanged messages.

For the distributed approach, we show results for MRPS with one round, three rounds and six rounds. These results illustrate the trade-off between solution quality and communication overhead. The growing threshold  $\alpha$  for a mission to release nodes in MRPS is set to 10% in the first round and is increased by 10% for each subsequent round until it reaches  $T$ , or 50%. Advertisement messages are sent from mission leaders to all nodes within two hops.

As an upper-bound we include the profit results for the optimal fractional solution, described in Section III. This is the optimal solution for the relaxed fractional problem in which sensors may divide their utility between multiple missions and all fractional profits are counted, regardless of whether the success threshold is not reached. Our algorithms' performance may be judged in comparison to this upper bound. We have also used brute force to find the optimal sensor assignments for several configurations. The difference between optimal and the centralized greedy scheme was between 0 and 1.2%.

**Results:** Figure 1 shows the fraction of the maximum mission profits achieved by the different schemes compared to the optimal fractional profits. The maximum profit is the sum of all missions profits. Note that when we create missions we do not guarantee that all of them can be satisfied by available sensors, i.e. some missions, due to network configuration, might not be satisfiable even if all available sensors are assigned to them.

The greedy centralized solution performs best followed, by the 6-round proposal. However, its advantage lessens as the density of the nodes increases. For same-sized fields with

1000 nodes deployed (not shown in the figures), all the curves except the one-round proposal are nearly aligned. This is expected since there are more sensors that can be assigned to the different missions. We note that the improvement in MRPS when going from a single round to 3 rounds is very pronounced. However, the improvement gained when jumping to 6 rounds is less apparent and may not justify the necessary communications overhead.

Figure 2 shows the communication overhead of the different schemes. As expected, the centralized scheme has the highest overhead. With MRPS, more messages are exchanged as the number of rounds increases. To study this effect, we fixed the number of nodes to 500 and varied the number of allowable rounds from 1 to 10. Not surprisingly, the achieved profits initially increase with the number of rounds. Beyond 8 rounds, however, the further increase is very small since in this setting most achievable missions have by then reached the success threshold. We also found that the communication overhead increases linearly with the number of rounds, as expected.

From the above results, we conclude that the distributed schemes perform well. The difference in achieved profits between centralized and distributed schemes was less than 8%. At the same time, it saved as much as 50% of the messages.

### C. Dynamic Scenarios

**Setup:** We test the performance of our distributed proposal scheme (DPS) in the dynamic setting (missions arrive and depart over time). The aforementioned assumptions apply here. Moreover, we assume that missions arrive according to a Poisson process. The mission lifetimes are selected according to an exponential distribution with an average lifetime of one hour and a maximum of four hours. The exponential distribution is heavy-tailed, which models realistic scenarios in which there are many short-lived missions and few long-lived ones. Although a centralized scheme is impractical in dynamic scenarios due to high communication cost, we include its results as a performance comparison. We rerun both the centralized scheme and the relaxed LP, on the current instance, upon each mission arrival or departure.

**Results:** We compare the performance of DPS to the greedy centralized scheme and the optimal fractional solution. Figure 3 shows a trace of the achieved network profits during a period of 12 hours with an arrival rate of  $\lambda = 6$  missions/hour, and a network of 500 nodes. We start recording the trace after 10

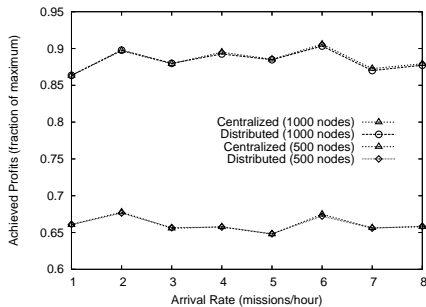


Fig. 4. Average achieved profits (per unit of time) in period of 50 hours

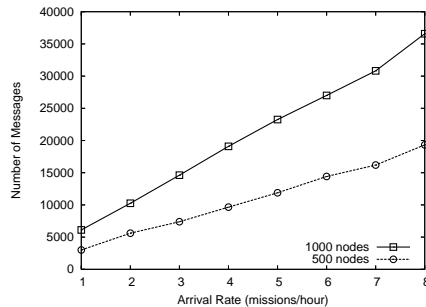


Fig. 5. Number of exchanged messages in period of 50 hours

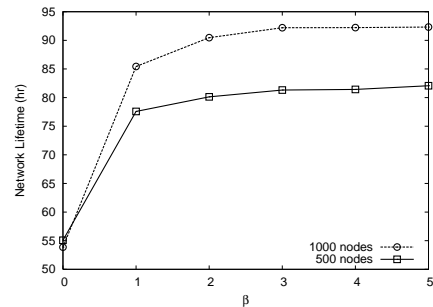


Fig. 6. Network lifetime (in hours)

hours. As can be seen, the performance of DPS is very close to that of the centralized scheme.

Figures 4 and 5 show the average performance over a period of 50 hours (averaged over 10 runs) for a network with 500 nodes and 1000 nodes. Figure 4 shows the average achieved profits per unit of time (fraction of maximum) as the mission arrival rate varies. We see that both the centralized and DPS perform almost equally. As expected, a network with more nodes achieves higher profits. The communication overhead of DPS is shown in Figure 5. The number of messages grows linearly as the number of missions increases. We omit message count for the centralized scheme as this value will necessarily be much larger than DPS's. This is because for each mission arrival and departure, the base station needs to collect information from all nodes that can contribute to the arriving mission to get status updates. The average number of messages exchanged per mission is around 40 for 500 nodes and 80 for 1000 nodes. This includes all the messages needed to advertise the mission and make all the assignment decisions.

Figure 6 shows the results for EDPS in network of 500 nodes and 1000 nodes (averaged over 10 runs). The mission arrival rate is set to 6 missions/hour. All nodes start with energy to support 10 hours of continuous sensing (we only consider energy consumed for sensing). Sensor reassignment is performed every 20 minutes to balance energy consumption. Choosing a smaller period may yield a more uniform assignment but would increase communication overhead.

We define the network lifetime as the time until the first node dies. Figure 6 shows the lifetime of the network for different values of the energy influence parameter  $\beta$ . Recall that when  $\beta = 0$ , EDPS becomes DPS. The results show that when EDPS is used, network lifetime increases by 50% and 70%, for networks with 500 and 1000 nodes, respectively. The increase is notable when  $\beta$  goes from 0 to 1, i.e. when we start taking energy into account. Beyond that point, the increase in lifetime is less pronounced. The denser the network, the more options the assignment scheme has in preventing node death, and hence the longer network is achievable.

The total achieved profits, which recall is the sum of profits in every timesteps over the network lifetime, also increases with lifetime increases (results omitted). The denser the network, the greater proportional increase in profits can be achieved by this method. Finally, due to periodic updates, the

communication overhead does increase when EDPS is used. We see an average increase of around 50% for both network sizes. Although the percentage increase may seem high, the actual numbers of exchanged messages per mission (around 60 for 500 nodes and 120 for 1000 nodes) are relatively small.

## VII. CONCLUSION

In this paper, we proposed a more general formulation of the problem of assigning sensors to missions and presented several solutions to solve it. We showed that it is strongly NP-Complete. As such, we turned to approximation algorithms and heuristics. We considered a greedy centralized algorithm and several distributed schemes, which we also adapted to the dynamic setting. Our simulation results demonstrate that under realistic conditions the greedy algorithm performs near-optimally and the distributed schemes perform almost as well.

**Acknowledgements.** This research was sponsored by US Army Research laboratory and the UK Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the US Government, the UK Ministry of Defence, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

## REFERENCES

- [1] A. Bar-Noy, T. Brown, M. Johnson, T. La Porta, O. Liu, and H. Rowaihy. Assigning sensors to missions with demands. In *ALGOSENSORS 2007*.
- [2] J. Byers and G. Nasser. Utility-based decision-making in wireless sensor networks. In *Proceedings of MobiHoc 2000*.
- [3] L. Kaplan. Global node selection for localization in a distributed sensor network. *IEEE Transactions on Aerospace and Electronic Systems*, 42(1):113–135, January 2006.
- [4] B. Karp and H. Kung. Greedy perimeter stateless routing for wireless networks. In *Proceedings of MobiCom 2000*.
- [5] M. Perillo and W. Heinzelman. Optimal sensor management under energy and reliability constraints. In *Proceedings of the IEEE Conference on Wireless Communications and Networking*, March 2003.
- [6] H. Rowaihy, M. Johnson, T. Brown, A. Bar-Noy, and T. La Porta. Assigning Sensors to Competing Missions (long version). Technical Report NAS-TR-0080-2007, Department of Computer Science and Engineering, Pennsylvania State University.
- [7] K. Shih, Y. Chen, C. Chiang, and B. Liu. A distributed active sensor selection scheme for wireless sensor networks. In *Proceedings of the IEEE Symposium on Computers and Communications*, June 2006.
- [8] F. Zhao, J. Shin, and J. Reich. Information-driven dynamic sensor collaboration. *IEEE Signal Processing Magazine*, 19(2):61–72, March 2002.