Maximising Sensor Network Efficiency Through Agent-Based Coordination of Sense/Sleep Schedules

A. Farinelli, A. Rogers and N. R. Jennings
School of Electronics and Computer Science
University of Southampton
Southampton, SO17 1BJ, UK.
{af2,acr,nrj}@ecs.soton.ac.uk

***Abstract*—In this paper we consider the problem of maximising the efficiency of a sensor network deployed for wide-area surveillance, by coordinating of the sense/sleep schedules of power constrained energy-harvesting sensor nodes. We propose a formal model of the wide-area surveillance problem that we face, and theoretically analyse the performance of a sensor network (i.e. the probability that an event within the environment is detected) in the case of (i) continuously powered, (ii) randomly coordinated, and (iii) optimal coordinated sensors. We show that coordinating the sense/sleep schedules of the sensors can yield a significant increase in the performance of the network. Hence, we demonstrate that we can appropriately decompose the system wide goal of maximising the probability that events are detected, in order that we can optimise it using generic decentralised agent-based coordination algorithms (specifically, one based on the max-sum algorithm) that use only local communication and computation. We empirically evaluate our approach in a simulated environment and show that this decentralised algorithm is able to successfully coordinate the sense/sleep schedule of sensors, while attaining results close to the theoretically indicated optimum.**

I. Introduction

The availability of small battery-powered wireless devices that can be deployed in the environment to acquire and integrate information opens the field to the use of wireless sensor networks for applications such as rescue robotics [1], wide-area surveillance [2] and environmental phenomena monitoring [3]. A key challenge for the successful deployment of such networks is to provide the individual sensors with the capability to perform efficient power management in order that they can meet the system wide requirements of maximising the lifetime of the sensor network while also collecting the maximum information possible.

To this end, recent work has begun to address the challenge of extending the lifetime of individual sensors by giving them the capability to harvest energy from their local environment; ideally using multiple sources such as solar cells or micro generators that can exploit vibrational energy or small temperature differences in combination [4]. Further work has shown that when equipped with sufficient energy harvesting resources, and the ability to model and predict future energy usage and harvesting, such sensors may then control their duty cycle (effectively switching between active sensing modes and low-power sleep modes) in order to operate in an *energy neutral* mode, and hence, exhibit unlimited lifetime [5].

However, in many application scenarios, the system wide performance of the network is dependent not only on the duty cycles of the individual sensors, but also on the interactions between these sensors. For example, within the 'Adaptive Energy-Aware Sensor Network' project that motivates this work, we aim to use energy-harvesting sensor nodes within a wide-area surveillance application in which events occurring in the environment should be detected by sensors randomly deployed across a wide geographical area. Such deployments typically exhibit redundancy whereby multiple sensors can observe the same portion of the environment. Thus, in order to maximise the efficiency of the sensor network (i.e. to maximise the probability with which events are detected), the individual sensors must not only control their own duty cycles (to satisfy their own individual energy constraints), but must also coordinate with neighbouring sensors (since energy is wasted when multiple sensors are actively sensing the same area at the same time). Furthermore, due to the random deployment of such sensors, and the fact that no single point has complete information regarding the position or state of each sensor, this coordination can not be performed *a priori*, nor in a centralised computation, but must be performed in a decentralised manner by the sensors themselves post deployment.

In the light of these requirements, a number of researchers have begun to tackle the challenge of coordinating the sense/sleep cycles of sensors within wireless sensor networks. For example, Hsin and Liu consider coordinating the duty cycles of non-energy harvesting sensors in order to maintain a minimum probability of event detection while maximising the lifetime of the individual sensors [6]. Giusti et al. consider the problem of coordinating the wake-up time of energy-neutral sensors, but do not explicitly consider the degree to which the sensing areas of neighbouring sensors overlap [7]. Conversely, Kumar et al. explicitly deal with the expected overlap of neighbouring sensors in a setting where each individual event must be detected by at least k sensors in order that it is identified correctly, but they do not provide a coordination mechanism. Rather, they provide guidance as to the number of sensors that should be deployed to achieve a specified degree of performance in the absence of any coordination [8]. Unfortunately, this earlier work does not address the specific wide-area surveillance problem that faces us here, and furthermore, the coordination approaches that are presented are problem specific and do not necessarily generalise to other settings.

In contrast, in this paper we propose an agent-based approach to satisfy local power management constraints while also addressing system wide performance goals. In doing so, we are able to exploit the extensive literature concerning the decentralised coordination of agent-based systems. Specifically, we consider each sensor to be represented by an agent with a state and a utility function. The state of the agent is the sense/sleep schedule of the sensor, and we decompose the system wide performance goal into individual utility functions for each agent, such that maximising the sum of these utility functions will also maximise the system wide goal.

Given this problem representation we can then apply generic agent-based coordination approaches to optimise the utility of the system in a decentralised way. In particular, we use the max-sum algorithm, a general coordination mechanism which was proposed as a suitable solution for distributed coordination in [9]. The max-sum algorithm allows agents to optimise a decomposable global function through distributed local computation and communication. The algorithm is able to compute solutions very close to the optimal, exhibits a low message overhead, is extremely robust to message failures, and is capable of being deployed on low-power sensing devices [9].

This paper builds on results obtained in [9], applying the max-sum algorithm for coordinating the sense/sleep cycle of sensors deployed for event detection. More specifically, we make the following contribution to the state of the art:

- We propose a formal model of the wide-area surveillance problem that we face. This model incorporates an ad hoc deployment of energy-neutral sensors using a fixed sense/sleep schedule, and we model the deployment of sensors, and the duration for which events are detectable as generic Poisson processes.
- We theoretically analyse the performance of the network in the case of (i) continuously powered, (ii) randomly coordinated, and (iii) optimally coordinated sensors, in order to give a theoretical upper bound on the performance gain that coordination could potentially yield. In a specific example we show that the optimally coordinated sensors detect 50% of the events that the randomly coordinated sensors fails to detect, or conversely, the optimally coordinated sensors are able to achieve the same level of performance as the randomly coordinated sensors using just 60% of their number.
- We show how we can appropriately decompose the system wide goal that we face in our wide-area surveillance problem (that of maximising the probability that an event is detected) into individual agent utility functions. Hence, we show that given this decomposition, we can apply the max-sum algorithm to optimise this system wide goal through local calculation and message passing.
- Finally, we evaluate our approach in a simulated environment. We benchmark the max-sum algorithm against two other coordination mechanisms (a centralised implementation of simulated annealing, and a decentralised solution using local best response). We show that max-sum is able to successfully coordinate the sensors through decentralised communication and computation, while attaining results very close to the optimal.

The remainder of this paper is structured as following: in section II we describe our formal model, before analysing it theoretically in section III. In section IV we introduce our agent-based coordination framework, and in section V we apply the max-sum algorithm to it. In section VI we present our empirical evaluations, and conclude in section VII.

II. Wide-Area Surveillance Problem

We now describe the formal model of the wide-area surveillance problem that we address in this paper. We assume that multiple sensors are deployed according to a Poisson process with rate per unit area λ_s (i.e. within a unit area we expect to find λ_s sensors)[1]. Each sensor has a circular sensing radius, r, and is tasked with detecting transient events within its sensing radius. We make no assumptions regarding the process by which events occur[2], and we consider a general case in which events may have a limited duration in which they remain detectable after their initial appearance. We describe this duration by an exponential distribution[3] such that the probability of an event lasting time t is $\lambda_d \mathrm{e}^{-\lambda_d t}$.

We assume that the sensors are able to harvest energy from their local environment, but at a rate that is insufficient to allow them to be powered continually. Thus at any time a sensor can be in one of two states: either sensing or sleeping. In the sensing state the sensor consumes energy at a constant rate, and is able to interact with the surrounding environment (e.g. it can detect events within its sensing radius and communicate with other sensors). In the sleep state the sensor can not interact with the environment but it consumes negligible energy. To maintain energy-neutral operation, and thus exhibit an unlimited lifetime, sensors adopt a duty cycle whereby within discrete time slots they switch between these two states according to a fixed schedule of length L. We denote the schedule of sensor i by a vector $\mathbf{s}_i = \{s_0^i, \ldots, s_{L-1}^i\}$ where $s_k^i \in \{0, 1\}$, and $s_k^i = 1$ indicates that sensor i is in its active sensing state during time slot k (and conversely, it is sleeping when $s_k^i = 0$). We assume that this schedule is repeated indefinitely, and here we consider schedules in which $\sum_{k=0}^{L-1} s_k^i = 1$ (i.e. the sensor is in its sense state for one of L time slots, and in a sleep state for all other time slots).

III. Theoretical Analysis

Given the model described above, we can theoretically analyse the performance of the network in three important cases, in order to calculate an upper bound on the performance gain that results from coordination:

[1]The use of Poisson processes to describe these events is common within the literature, and represents a generic, non-domain specific model [6].

[2]Our model is independent of assumptions regarding the process by which event occur, and is not limited to those uniformly distributed in space or time, as long as we have no prior belief as to when and where events may occur.

[3]This exponentially distributed duration naturally arises from a Poisson process that describes the departure or disappearance of the events.

1) **Continuously Powered Sensors**:
We initially ignore the energy constraints of the sensors and assume that they remain in their sensing state continuously. This represents an absolute upper bound on the performance of the network.
2) **Randomly Coordinated Sensors**:
We assume that the sensors are indeed limited to sensing for just one in every L time slots, and that the choice of which time slot to use is made randomly by each individual sensor with no coordination with nearby sensors.
3) **Optimally Coordinated Sensors**:
As above we again consider sensors limited to sensing for just one in every L time slots, but we consider that they are able to optimally coordinate the choice of sensing time slot with neighbouring sensors whose sensing radius overlaps with their own.

In each case, we calculate the efficiency, E, of the sensor network; expressed as the probability that any event that occurs within the environment is indeed detected by the network.

A. Continuously Powered Sensors

If we assume that the sensors remain continuously in their sense state then an event will be detected if it occurs within the sensing radius of at least one sensor. Given the Poisson process that describes the deployment of sensors, the probability that an event falls within the sensing radius of m sensors is given by $(\lambda_s \pi r^2)^m \mathrm{e}^{-\lambda_s \pi r^2}/m!$. Thus, an event to be detected in all cases that $m > 0$, and thus, the efficiency is given by:

$$E_{\text{continuous}} = 1 - \mathrm{e}^{-\lambda_s \pi r^2} \quad (1)$$

Clearly, increasing either the density of the sensor, λ_s, or the sensing radius of the sensors, r, increases the probability with which events are detected.

B. Randomly Coordinated Sensors

In order to calculate the efficiency of the network when sensors are energy constrained, and use a sensing schedule in which one time slot is independently randomly selected for sensing, we first consider the more general case of an area that falls within the sensing radius of multiple sensors. The effective sensing schedule of this area is simply given by the logical OR of the schedules of each individual sensor and is described by the vector $\mathbf{s} = \{s_0, \dots, s_{L-1}\}$. An event will be detected within this area, if (i) it occurs while one of the sensors is currently in its sense state, or (ii) if it occurs when all sensors are sleeping, but is still detectable when one sensor starts sensing again.

Given the exponential distribution for the time during which an event remains detectable after its initial occurrence, the probability of an event being detectable after time t is given by $\int_t^{\infty} \lambda_d \mathrm{e}^{-\lambda_d \tau} d\tau = \mathrm{e}^{-\lambda_d t}$. Thus, if we consider that an event occurs within any specific time slot, and define n as the number of time slots until one of the sensors is again in its sensing state (where $n = 0$ indicates that one of the sensors is currently in its sense state), then the probability

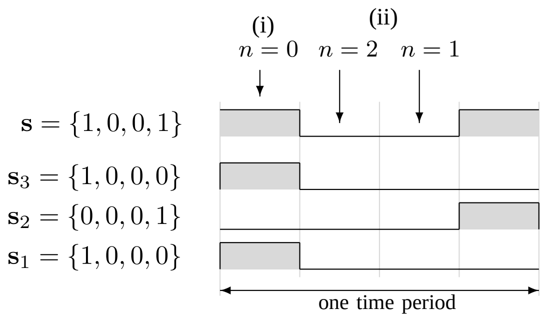
(i)
(ii)
$n=0$
$n=2$
$n=1$
$\mathbf{s} = \{1,0,0,1\}$
$\mathbf{s}_3 = \{1,0,0,0\}$
$\mathbf{s}_2 = \{0,0,0,1\}$
$\mathbf{s}_1 = \{1,0,0,0\}$
one time period

Fig. 1. Example showing the effective schedule for an area that falls within the sensing radius of three sensors with sensing schedules $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$.

of detecting the event is 1 when $n = 0$, and is given by $L \int_0^{1/L} \mathrm{e}^{-\lambda_d (n/L - t)} dt = \frac{\mathrm{e}^{-\lambda_d n/L} L}{\lambda_d} \left(\mathrm{e}^{\lambda_d/L} - 1 \right)$ when $n \geq 1$. Figure 1 illustrates both cases for an area that falls within the sensing radius of three sensors with sensing schedules $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$.

This result is used in Algorithm 1 to calculate the probability that an an event is detected if it occurs within an area whose sensing schedule is described by the vector $\mathbf{s} = \{s_0, \dots, s_{L-1}\}$. Note that as λ_d increases (such that the events become increasingly transient), then the probability of detection decreases toward only detecting the event during the cycle in which the sensor is in its sense state. Conversely, as λ_d decreases toward zero (such that the events become increasingly long lived), then the probability of detecting the event approaches one.

We can then use this result to calculate the probability of detecting an event assuming that each sensor individually selects one of the L time slots in which to sense. We do so by summing over the probabilities that any point in the environment is within the sensing radius of m sensors, and that the sensing schedules of these m sensors combine to give any of the 2^L possible sensing schedules (denoted by $\mathcal{S}$). In the latter case, the probability of any sensing schedule, $\mathbf{s}$, arising from the combination of m individual schedules, each of length L with a single active sensing time slot, is given by $\sum_{k=0}^{n} (-1)^k \binom{n}{k} (n-k)^m / L^m$, where n is the number of sensing time slots in the combined schedule[4]. Algorithm 2 shows this calculation in pseudo-code.

C. Optimally Coordinated Sensors

Finally, we can calculate an upper bound for the effectiveness of coordination between sensors. To do so, we assume that if any point in the environment is within the sensing radius of m sensors, then these sensors are able to perfectly coordinate their sensing schedules in order to maximise the probability that an event is detected at this point. This represents a strict upper bound on the efficiency of the network, since we ignore the real constraints on achieving this coordination for

[4]Note that the numerator in this expression is a standard result in probability theory regarding the number of ways in which m balls may be placed into L cups such that exactly n of them are not empty, and the denominator is the total number of ways in which m balls may be placed in L cups.

Algorithm 1 $P(\text{detection}|\lambda_d, \mathbf{s})$

```
1:  $value \leftarrow 0$
2:  for $i = 0$ to $L - 1$ do
3:      $n \leftarrow 0; j \leftarrow i$
4:      while $s_j = 0$ do
5:          $j \leftarrow \text{mod}(j + 1, L); n \leftarrow n + 1$
6:      end while
7:      if $n = 0$ then
8:          $value \leftarrow value + 1/L$
9:      else
10:         $value \leftarrow value + \mathrm{e}^{-\lambda_d n/L}\left(\mathrm{e}^{\lambda_d/L} - 1\right)/\lambda_d$
11:     end if
12: end for
13: return $value$
```

Algorithm 2 $E_{\text{random}}(\lambda_s, \lambda_d, r, L)$

```
1:  $value \leftarrow 0$
2:  for $m = 1$ to $\infty$ do
3:      $P(m) = (\lambda_s \pi r^2)^m \mathrm{e}^{-\lambda_s \pi r^2}/m!$
4:      for $\mathbf{s} \in \mathcal{S}$ do
5:          $n \leftarrow \sum_{k=0}^{L-1} s_k$
6:          if $n \leq m$ then
7:              $P(\mathbf{s}) = \sum_{k=0}^{n}(-1)^k\binom{n}{k}(n-k)^m/L^m$
8:              $value \leftarrow value + P(m) \times P(\mathbf{s}) \times P(\text{detection}|\lambda_d, \mathbf{s})$
9:          end if
10:     end for
11: end for
12: return $value$
```

Algorithm 3 $E_{\text{optimal}}(\lambda_s, \lambda_d, r, L)$

```
1:  $value \leftarrow 0$
2:  for $m = 1$ to $\infty$ do
3:      $P(m) = (\lambda_s \pi r^2)^m \mathrm{e}^{-\lambda_s \pi r^2}/m!$
4:      if $m < L$ then
5:          $value \leftarrow value + P(m) \times P(\text{detection}|\lambda_d, \mathbf{s}^*_{m,L})$
6:      else
7:          $value \leftarrow value + P(m)$
8:      end if
9:  end for
10: return $value$
```

any given sensor network configuration[5]. Thus, if $m \geq L$ we assume that the area is continually sensed, and when $1 < m < L$ we assume an optimal sensing schedule, $\mathbf{s}^*_{m,L}$, that can be automatically derived through exhaustive search using algorithm 1, or more simply, by noting that the detection probability is maximised when the schedule contains m sensed time slots that are maximally separated. For example, if $L = 4$, then $\mathbf{s}^*_{1,4} = \{1, 0, 0, 0\}$, $\mathbf{s}^*_{2,4} = \{1, 0, 1, 0\}$ and $\mathbf{s}^*_{3,4} = \{1, 1, 1, 0\}$. Algorithm 3 shows this calculation, and in Section VI we show that a centralised solution to the optimisation of the sensor network does in fact closely approximate this upper bound.

D. Network Performance Comparison

Using the three theoretical results presented in this section we can calculate the maximum gain in system wide performance that coordination may yield. Figure 2 shows an example of this calculation in the case that sensors may only actively

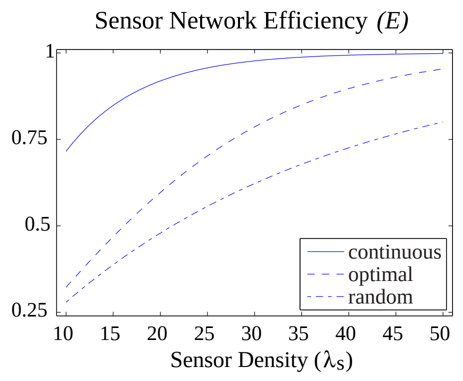

Fig. 2. Theoretical results showing sensor network efficiency for three different coordination cases ($L = 4$, $r = 0.2$ and $\lambda_d >> 1/L$.)

sense for 1 in every 4 time slots (i.e. $L = 4$), and the departure rate of events is much greater than $1/L$ (i.e. events are very short lived)[6]. The plot shows the efficiency of the network when all sensors are continuously sensing (the absolute upper bound on performance), when they randomly coordinate their sense/sleep cycles, and when they coordinate their sense/sleep cycles optimally. Clearly, as the density of the sensor deployment, λ_s, increases then the overall efficiency of the network increases, and in the limit, all events that occur within the environment are detected. Furthermore, note that the optimally coordinated network always out performs the randomly coordinated network, and as the density of the deployment increases, the gain increases. Indeed, in this example, when $\lambda_s > 35$ the optimally coordinated network detects 50% of the events that the randomly coordinated network fails to detect, or conversely, the optimally coordinated network is able to achieve the same level of performance as the randomly coordinated network with just 60% of the sensors deployed. Similar results are observed for other values of L, with coordination having great potential benefit as L increases.

IV. Agent Based Coordination

The above results indicate that significant gains can be realised through coordination in this setting. However, they do not indicate whether or not these gains can be achieved using a decentralised coordination mechanism. In previous work, domain specific solutions have been proposed to address similar problems [6], [7], [8]. However, here we advocate a general agent based approach to satisfy local constraints while also addressing system wide performance goals.

We first note that since sensors can not communicate when they are in sleep mode, a natural way to address the coordination problem is to have an initial negotiation phase in which all the sensors are active, and exchange messages to devise a sense/sleep schedule that they will maintain for a fixed time [7]. This negotiation phase can be repeated to account for possible sensor failures or topology changes. However, the

[5]Note that this is equivalent to the statement that zero clashes is a strict lower bound for solutions to a graph colouring problem, even though in any specific problem instance may not be colourable.

[6]As discussed in Section III-B, this represents the lower limit of performance of the network, where events are very short lived, and can only be detected if a sensor is in its active sensing state when the event occurs. It also represents the case where coordination can have the most significant impact.

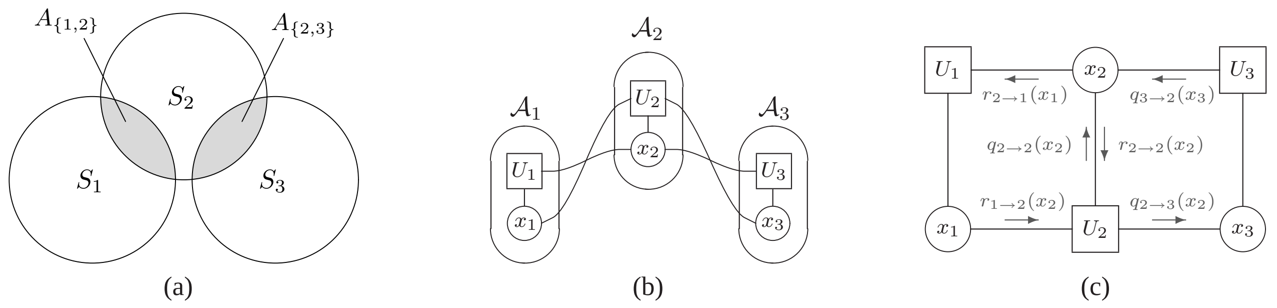

Fig. 3. Diagram showing (a) the position of three sensors in the environment, (b) the resulting factor graph with agents representing each sensors, and (c) subset of the messages exchanged over the factor graph using the max-sum algorithm.

frequency with which the negotiation phase is repeated will be problem specific, and thus, we do not address it here.

Hence, to apply agent-based coordination techniques within this negotiation phase, we must first decompose the system wide goal that we face (that of maximising the probability that an event is detected) into individual agent utility functions. We can then directly apply a number of algorithms that seek to maximise the sum of the utilities of all agents in the system, through local computation and communication.

To this end, we represent each sensor as an agent, and decompose this agent into a function and a variable that represent its state and utility. The utility of any agent represents the probability that it will detect an event within the environment, and the state of the agent $x_i \in \mathbf{X}$ represents the sense/sleep schedule that it has selected. The utility of the agent is thus dependent on its own state, and also on the state of the agents that represent neighbouring sensors whose sensing radii overlap with its own. Such interactions can conveniently be represented as a *factor graph* containing variable (state) and function (utility) nodes [9]. Figures 3(a) and 3(b) show an example of this decomposition in which three sensors, $\{S_1, S_2, S_3\}$, are represented by three agent $\{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3\}$ with utilities $\{U_1, U_2, U_3\}$ and states $\{x_1, x_2, x_3\}$.

Now, given a set $\mathbf{N}$ of agents (sensors) that form a sensor network, each of which can select a state from a discrete set of options (equivalent to selecting a sensing schedule), our goal is to find the optimal state of each agent, $\mathbf{x}^*$, such that the sum of the utilities of the agents is maximised:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \sum_{i=1}^{|\mathbf{N}|} U_i(\mathbf{x}_i) \tag{2}$$

where $\mathbf{x}_i$ is a vector representing the states of all the agents that determine the utility of agent i. For example, from Figure 3(b) we have $\mathbf{x}_1 = \{x_1, x_2\}$, $\mathbf{x}_2 = \{x_1, x_2, x_3\}$ and $\mathbf{x}_3 = \{x_2, x_3\}$.

The utility of each agent is determined by its own sense/sleep schedule, and by those of agents whose sensing radii overlap with its own. This can easily be determined by each agent assuming that it knows the relative positions of these other agents (a strong assumption common in the literature [6], and one that we intend to relax in our future work). Thus, we define $\mathbf{N}_i$ to be a set of indexes indicating which other agents' sensing radii overlap with that of agent i (again from Figure 3(a) we have $\mathbf{N}_1 = \{2\}$, $\mathbf{N}_2 = \{1, 3\}$ and $\mathbf{N}_3 = \{2\}$) and $\mathbf{k}$ is any subset of $\mathbf{N}_i$ (including the empty set). $A_{\{i\}\cup\mathbf{k}}$ is the area that is overlapped only by agent i and those agents in $\mathbf{k}$. For example, with respect to Figure 3(a), the area $A_{\{1,2\}}$ is the area that is sensed only by agents 1 and 2.

We define a function $G : 2^{\mathbf{X}} \to \mathcal{S}$ such that $G(\mathbf{x}_{\{i\}\cup\mathbf{k}})$ is the combined sensing schedule of agent i and those agents in $\mathbf{k}$ (calculated through the logical 'OR' of each individual schedule, as shown in Figure 1). The utility of agent i is then given by:

$$U_i(\mathbf{x}_i) = \sum_{\mathbf{k}\subseteq\mathbf{N}_i} \frac{A_{\{i\}\cup\mathbf{k}}}{|\{i\}\cup\mathbf{k}|} \times P(\text{detection}|\lambda_d, G(\mathbf{x}_{\{i\}\cup\mathbf{k}})) \tag{3}$$

where $P(\text{detection}|\lambda_d, G(\mathbf{x}_{\{i\}\cup\mathbf{k}}))$ is given by Algorithm 1. Note, that we scale the area by the number of agents who can sense it to avoid double-counting areas which are represented by multiple sensors. Also, note that when the set $\mathbf{k}$ is empty we consider the area covered only by the single sensor.

V. The Max-Sum Approach to Coordination

The decomposition described above is very general, and in practice, we can now apply any agent-based decentralised coordination algorithm to determine the individual sense/sleep schedules that maximise the sum of the agents' utilities. In this paper, we use a message passing technique based upon the max-sum algorithm. The rationale for this choice is that this technique represents an ideal combination of the best features of complete algorithms and approximate stochastic algorithms. It can make efficient use of constrained computational and communication resources, and yet it is able to effectively represent and communicate complex utility relationships through the network and attain close to optimal solutions. Such message passing techniques are widely used within graphical model for belief propagation [10], [11], and the max-sum algorithm has previously been demonstrated to produce decentralised solution to graph colouring problems (a canonical coordination problem very similar to that which faces us here) that are close to the optimal. The algorithm has a low communication overhead, is robust to message failures, and has been demonstrated on low-power devices [9].

The max-sum algorithm operates directly on the factor graph representation of the global function to be optimised (as shown in Figure 3(b)). When this graph is cycle free, the algorithm is guaranteed to converge to the global optimal solution of the welfare maximisation problem (i.e. it finds the combination of states that maximises the sum of the agents' utilities). When applied to cyclic graphs (as is the case here), there is no guarantee of convergence but extensive empirical evidence demonstrates that this family of algorithms generate very good approximate solutions [10], [11].

The max-sum algorithm solves this problem in a decentralised manner by specifying messages that should be passed from variable to function nodes, and from function to variable nodes. These messages are defined as:

From variable to function:

$$q_{i\to j}(x_i) = \alpha_{ij} + \sum_{k\in\mathcal{M}_i\setminus j} r_{k\to i}(x_i) \tag{4}$$

where α_{ij} is a scaler chosen such that $\sum_{x_i} q_{i\to j}(x_i) = 0$, and $\mathcal{M}_i$ is a vector of function indexes, indicating which function nodes are connected to variable node i.

From function to variable:

$$r_{j\to i}(x_i) = \max_{\mathbf{x}_j\setminus i}\left[U_j(\mathbf{x}_j) + \sum_{k\in\mathcal{N}_j\setminus i} q_{k\to j}(x_k)\right] \tag{5}$$

where, $\mathcal{N}_j$ is a vector of variable indexes, indicating which variable nodes are connected to function node j and $\mathbf{x}_j\setminus i \equiv \{x_k : k\in\mathcal{N}_j\setminus i\}$.

At any time during the propagation of these messages, agent i is able to calculate the marginal function, $z_i(x_i)$, from the messages flowing into agent i's variable node:

$$z_i(x_i) = \sum_{j\in\mathcal{N}_i} r_{j\to i}(x_i) \tag{6}$$

Then by computing $\arg\max_{x_i} z_i(x_i)$, each individual agent is able to determine which state it should adopt such that the sum of the agents' utilities is maximised. Note that although the max-sum algorithm is approximating a global optimisation problem (in this case the coordination of sensor sense/sleep schedules) it involves only local communication and computation.

Figure 3(c) shows a subset of the messages for the factor graph of Figure 3b. To better illustrate the functioning of the max-sum algorithm we can consider the computation of a sample variable to function message, $q_{2\to 3}(x_2)$, and a sample function to variable message, $r_{2\to 1}(x_1)$. For ease of exposition we ignore the scaler α_{23} and then using Equation 4 the message $q_{2\to 3}(x_2)$ is given by:

$$q_{2\to 3}(x_2) = \sum_{k\in\mathcal{M}_2\setminus 3} r_{k\to 2}(x_2)$$

Considering that in our case $\mathcal{M}_2\setminus 3 = \{1,2\}$, then we can expand the summation to obtain:

$$q_{2\to 3}(x_2) = r_{1\to 2}(x_2) + r_{2\to 2}(x_2)$$

Therefore, for variable to function messages each agent only needs to aggregate the information received from all the neighbouring functions, without considering the receiver of the message (i.e. function node of agent 3 in our case). The aggregation is performed simply by summing the messages. Notice that messages in the max-sum algorithms are not values, but functions. In our case, since the agents have a discrete variable state, we represent each message as a vector with d components where d is the number of states the variable can take on. When we consider a sensing schedule of length L such that $\sum_{k=0}^{L-1} s_k^i = 1$, we consider that there are L discrete possible schedules (each corresponding to the time slot in which the sensor is actively sensing), and thus, $d = L$. In more complex setting, with heterogeneous sensing schedules across sensors, we can simply use a more complex mapping between the state and the schedule.

As for the message $r_{2\to 1}(x_1)$ from Equation 5 we have:

$$r_{2\to 1}(x_1) = \max_{\mathbf{x}_2\setminus 2}\left[U_2(\mathbf{x}_2) + \sum_{k\in\mathcal{N}_\mathbf{2}\setminus 1} q_{k\to 2}(x_k)\right]$$

and considering that in our case $\mathcal{N}_2\setminus 1 = \{2,3\}$ and that $\mathbf{x}_2\setminus 1 = \{x_2, x_3\}$ we then have:

$$r_{2\to 1}(x_1) = \max_{x_2,x_3}\left[U_2(x_1,x_2,x_3) + q_{2\to 2}(x_2) + q_{3\to 2}(x_3)\right]$$

Therefore, for function to variable messages each agent needs to maximise a function which results from the summation of its utility and the messages received from neighbouring variables. The maximisation step is the one that requires more computational effort. Since the variables are discrete, we can perform this maximisation step simply computing the above function for all the possible values of the argument of the message (i.e. x_1 in our sample message) maximising over all the possible combination of the variables that are the argument of the maximisation (i.e. x_2 and x_3 in our sample message). Note that this computation is exponential only in the number of neighbours. This is typically much less than the total number of agents within the system, and thus the algorithm scales well as the number of agents is increased.

The messages described above may be randomly initialised, and then updated whenever an agent receives an updated message from a neighbouring agent; there is no need for a strict ordering or synchronisation of the messages. In addition, the calculation of the marginal function shown in Equation 6 can be performed at any time (using the most recent messages received), and thus, agents have a continuously updated estimate of their optimum state. In our application

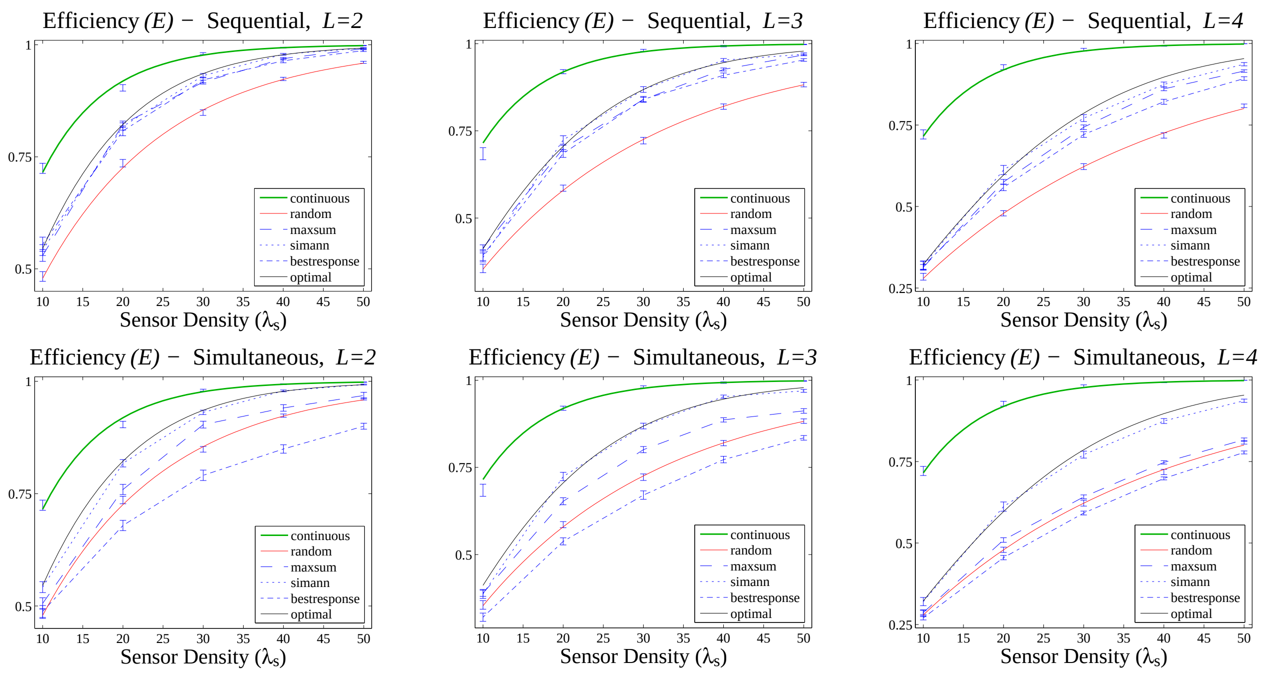

Fig. 4. Coverage value against lambda where $r = 0.20$, $\lambda_d = 20$ varying L (from 2 to 4), and the update model (sequential or simultaneous)

scenario, this is not necessary, and we continue to propagate messages throughout the initial negotiation phase, and then compute the optimum state.

As for communication overhead, the max-sum exchanges messages of fixed size, where the size of each message is dependent on the domain of the variable[7]. The number of messages that the agents need to exchange increases only linearly with the number of neighbours and the number of algorithm iterations. As for the number of iterations, max-sum usually is able to compute solutions of good quality in few cycles (see [9] for a quantitative evaluation of convergence). In any case, the number of iteration of the algorithm can be tuned by the system designer addressing the trade-off between solution quality and energy consumption.

VI. Experimental Evaluation

To evaluate our approach we have developed a Java based simulation environment where sensors can be deployed on a planar Cartesian space according to the Poisson process described in Section II. Each sensors is represented by an agent that exchanges messages with its neighbours during the negotiation phase in order to devise a coordinated sense/sleep schedule. Given the sensor deployment and the sensor schedules, the efficiency of the system is then evaluated. To simplify the experimental setting we consider homogeneous sensing ranges and power constraints for all sensors. However, we note that these are not requirements of our approach.

For the initial negotiation phase, in addition to the max-sum algorithm described in the last section we implement and compare two additional coordination algorithms:

1) **Local Best Response**: Rather than apply the max-sum message passing algorithm, sensors simply exchange message indicating their preferred sense/sleep schedule, and this preferred schedule is calculated to be the best response to that of the overlapping sensors (based on earlier messages received from these sensors).
2) **Centralised Simulated Annealing**: Rather than perform decentralised optimisation, we implement a centralised version of simulated annealing. This algorithm has complete knowledge of the topology of the network, and acts as a practical upper bound for the effectiveness of coordination given any particular sensor network (and as a comparison between the theoretical upper bound calculated in Section III-C).

Furthermore, during the initial negotiation phase we consider two different update models:

1) **Sequential Update**: Only one sensor executes its coordination algorithm while the others are waiting. The messages that have been sent by the executing sensor are received by all the other neighbouring sensors before they start their execution phase. The order in which agents execute is randomised at each time step. This setting models a situation where computation and communication are nearly instantaneous.
2) **Simultaneous Update**: All sensors execute their coordination algorithm at the same time, and they receive messages from the other sensors only when all sensors have completed this computation. This setting models a situation where communication is significantly slower than computation.

Finally, we also simulate the two results calculated in Section III. Specifically, the case where sensors randomly coordinate, and when they remain in their sensing state continuously.

We evaluate each case for a range of sensor densities, λ_s, and schedule lengths, L, and in each case perform 100

[7]In particular in our case, the size of each message is L.

repetitions over randomly deployed sensor networks. The results of these evaluations are shown in Figure 4, where the error bars indicate the mean and the standard error in the mean.

First note that the simulation and theoretical results for the continuously powered and randomly coordinated sensors agree perfectly, confirming the accuracy of the simulation. Second, note that the centralised simulated annealing solution closely approximates the theoretical optimal coordination case calculated in Section III-C, indicating that this is a useful theoretical result since it closely reflects what is possible in practice under ideal conditions.

The results of the max-sum and best response algorithms are clearly dependent upon the update model. Considering the sequential update model first (top row of Figure 4), we see that both max-sum and best response attain values which are very close to the optimal. This is an interesting result since while there is much empirical evidence for the good performance of the max-sum algorithm, best response will clearly become trapped in local maxima, and thus, potentially provide results arbitrarily far from the optimal. The behaviour of best response in our domain suggests that for this particular problem local maxima are not a major problem from a practical point of view. Note also that as the sensor spends longer sleeping (i.e. as L increases), then the performance of best response (while still acceptable) decrease. However, max-sum is still very close to the optimal solution, and thus, becomes increasingly superior (since increasing L corresponds to an increase in the size of the search space, and thus local maxima have a more significant impact on the performance of best response).

As for the simultaneous update model (bottom row of Figure 4) it is possible to see that in this setting best response performs very poorly (i.e. worst than random). This is a well known problem for best response techniques; all sensors simultaneous computes the best response to their neighbours, and by updating at the same time, find themselves in another conflicting setting. Note that the max-sum algorithm is much less affected by this, since decisions do not depend directly on the preferred state of neighbouring sensors, but rather on messages that reflect longer range interactions.

VII. Conclusions and Future Work

The empirical results presented above indicate that the theoretical benefits of coordination derived in Section III can actually be realised in practice through the use of decentralised agent-based coordination algorithms (specifically one based upon the max-sum algorithm). Our future work consists of relaxing some of the strong assumptions that we have made regarding how each individual sensor calculates the probability that it will detect an event. Specifically, we wish to relax the assumptions that sensors have regular circular sensing areas, that they are able to determine the area of overlap with neighbouring sensors, and that events occur uniformly distributed over these areas. In reality, sensors may have highly irregular and obscured sensing areas, they may not be able to determine the exact position of neighbouring sensors, and events may be more likely to occur in some area than others. Thus, rather than calculate the probability of detecting an event based on an *a priori* model of the environment, we intend to investigate algorithms that are capable of learning the probability with which two sensors will observe the same event, if both are actively sensing at the same time. A simple and effective way to do this is to have an additional *calibration phase* before the negotiation phase, in which individual sensors are all active and learn mutual relationships through observing and exchanging information regarding events in the environment. This will typically require events to be identifiable (e.g. by their times of arrival, positions or other characteristics). Within the data fusion and tracking literature, this problem is commonly known as data or track association and various techniques have been developed to address such issues. Notice that in the calibration phase we are not interested in accurate tracking, we need only a reasonable estimates of the utility that a set of sensors will gain by being powered at different times. Finally, note that this change in the way the utility of the agent is calculated has no affect on the operation of the agent-based algorithm used to actually coordinate the choice of sense/sleep schedule, and we believe that the decoupling of the two is a key advantage of our approach.

VIII. Acknowledgments

The work reported on here was undertaken as part of the Data Information Fusion Defence Technology Centre (DIF DTC) Phase II 'Adaptive Energy-Aware Sensor Networks' project and is joint funded by MoD and General Dynamics UK Ltd.

References

[1] P. Rybski, S. Stoeter, M. Gini, D. Hougen, and N. Papanikolopoulos, "Effects of limited bandwidth communications channels on the control of multiple robots," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2001, pp. 369–374.

[2] A. Makarenko and H. Durrant-Whyte, "Decentralized data fusion and control algorithms in active sensor networks," in *Proc. of Seventh International Conference on Information Fusion*, 2004, pp. 479–486.

[3] J. K. Hart and K. Martinez, "Environmental Sensor Networks: A revolution in the earth system science?" *Earth-Science Reviews*, vol. 78, pp. 177–191, 2006.

[4] A. S. Weddell, N. R. Harris, and N. M. White, "Alternative Energy Sources for Sensor Nodes: Rationalized Design for Long-Term Deployment," in *Proc. of the IEEE International Instrumentation and Measurement Technology Conference (I^2MTC)*, 2008, in press.

[5] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, "Power management in energy harvesting sensor networks," *ACM Transactions on Embedded Computing Systems*, vol. 6, no. 4, 2007.

[6] C. Hsin and M. Liu, "Network coverage using low duty-cycled sensors: Random & coordinated sleep algorithm," in *Proc. of the Third International Symposium on Information Processing in Sensor Networks (IPSN)*, 2004, pp. 433–442.

[7] A. Giusti, A. L. Murphy, and G. P. Picco, "Decentralised scattering of wake-up times in wireless sensor networks," in *Proc. of the Fourth European Conference on Wireless Sensor Networks*, 2007, pp. 245–260.

[8] S. Kumar, H. T. Lai, and B. J., "On k-coverage in a mostly sleeping sensor network," in *Proc. of the Tenth Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2004, pp. 144–158.

[9] A. Farinelli, A. Rogers, A. Petcu, and N. R. Jennings, "Decentralised coordination of low-power embedded devices using the max-sum algorithm," in *Proc. of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2008, in press.

[10] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 42, no. 2, pp. 498–519, 2001.

[11] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.