Self-organising Sensors for Wide Area Surveillance Using the Max-sum Algorithm

Alex Rogers[1], Alessandro Farinelli[2], and Nicholas R. Jennings[1]

[1] School of Electronics and Computer Science
University of Southampton, Southampton, SO17 1BJ, UK
{acr,nrj}@ecs.soton.ac.uk
[2] Department of Computer Science
University of Verona, Verona, Italy
alessandro.farinelli@univr.it

Abstract. In this paper, we consider the self-organisation of sensors within a network deployed for wide area surveillance. We present a decentralised coordination algorithm based upon the max-sum algorithm and demonstrate how self-organisation can be achieved within a setting where sensors are deployed with no *a priori* information regarding their local environment. These energy-constrained sensors first learn how their actions interact with those of neighbouring sensors, and then use the max-sum algorithm to coordinate their sense/sleep schedules in order to maximise the effectiveness of the sensor network as a whole. In a simulation we show that this approach yields a 30% reduction in the number of vehicles that the sensor network fails to detect (compared to an uncoordinated network), and this performance is close to that achieved by a benchmark centralised optimisation algorithm (simulated annealing).

1 Introduction

The vision of computational systems composed of multiple autonomous interacting agents has been a research aim for at least two decades now, and is increasingly being realised in the real-world by the deployment of wireless sensor networks [1]. Such networks have found application in wide-area surveillance, animal tracking, and for monitoring environmental phenomena in remote locations, and a fundamental challenge within all such applications arises due to the fact that the sensors within these networks are often deployed in an ad-hoc manner (e.g. dropped from an aircraft or ground vehicle within a military surveillance application). In this case, the local environment of each sensor, and hence the exact configuration of the network, cannot be determined prior to deployment, and thus, the sensors themselves must be equipped with capability to *self-organise* sometime after deployment once the local environment in which they (and their neighbours) find themselves has been determined. Examples of such self-organisation include determining the most energy-efficient communication paths within the network once the actual reliability of communication links between individual sensors can be measured in situ, determining the optimal

D. Weyns et al. (Eds.): SOAR 2009, LNCS 6090, pp. 84–100, 2010.

orientation of sensors to track multiple moving targets that move through the sensor network, and in the application that we consider in detail in this paper, coordinating the sense/sleep schedules (or duty cycles) of power constrained sensors deployed in a wide-area surveillance task once the degree of overlap of the sensing fields of nearby sensors has been determined.

A common feature of these self-organisation problems is that the sensors must typically choose between a small number of possible states (e.g. which neighbouring sensor to transmit data to, or which sense/sleep schedule to adopt), and the effectiveness of the sensor network as a whole depends not only on the individual choices of state made by each sensor, but on the joint choices of interacting sensors. Thus, to maximise the overall effectiveness of the sensor network, the sensors within the network must typically make coordinated, rather than independent, decisions. Furthermore, this coordinated decision must be performed despite the specific constraints of each individual device (such as limited power, communication and computational resources), and the fact that each device can typically only communicate with the few other devices in its local neighbourhood (due to the use of low-power wireless transceivers, the small form factor of the device and antenna, and the hostile environments in which they are deployed). Additional challenges arise through the need to perform such coordination in a decentralised manner such that there is no central point of failure and no communication bottleneck, and to ensure that the deployed solution scales well as the number of devices within the network increases.

Problems of this nature are often described within the multi-agent systems literature as *Distributed Constraint Optimisation Problems* (DCOPs), and are often represented by graphs in which the nodes represent the agents (in this case the sensors), and edges represent real valued constraints that arise between the agents depending on their combined choice of state. A number of distributed complete algorithms have been proposed that allow the agents within the graph to make coordinated decisions over their states (communicating only with other agents who are connected to themselves through a constraint) such that the sum over all of the constraints is maximised (or minimised). An example of such an algorithm is DPOP which preprocess the constraint graph, arranging it into a *Depth First Search* (DFS) tree, before exchanging messages over this tree [2]. However, complete algorithms calculate the globally optimal solution, and such optimality demands that some aspect of the algorithm grows exponential in size. For example, within DPOP, the size of the messages is exponential in the width of the tree. Such exponential relationships are simply unacceptable for the constrained devices deployed within typical sensor networks.

In contrast, a large number of approximate algorithms have also been proposed for solving DCOPs. These algorithms are typically based upon entirely local computation, whereby each agent updates its state based only on the communicated (or observed) states of those local neighbours that influence its utility. As such, these approaches are well suited for large scale distributed applications, and in this context, the Distributed Stochastic Algorithm (DSA), is one of the most promising having been proposed for decentralised coordination within

sensor networks [3]. However, algorithms of this type often converge to poor quality solutions since the agents do not explicitly communicate their utility for being in any particular state, but only communicate their preferred state (i.e. the one that will maximise their own utility) based on the current preferred state of their neighbours. Furthermore, DSA introduces a parameter (which represents the probablity that an agent should actually update its state if the states of its neighbours indicate that this would be approapriate) whose value must be determined through trial and error prior to deployment of the algorithm.

Thus, against this background, there is a clear need for decentralised coordination algorithms that make efficient use of the constrained computational and communication resources found within wireless networks sensor systems, and yet are able to effectively represent and communicate complex utility relationships between sensors. To address this shortcoming, in this paper, we describe an approximate decentralised coordination algorithm that can be applied to the general problem of maximising the overall effectiveness of a sensor network (or any other decentralised system) in which the utility of any individual sensor is dependent not only on its own state, but also on the state of a number of interacting neighbours (as is the case in the example applications we have discussed previously). Our solution is based upon a novel representation of the problem as a cyclic bipartite factor graph and exploits message passing techniques that are frequently used in the context of information theory to decompose complex computations on single processors. We have previously demonstrated the effectiveness of this approach on a number of benchmark graph-colouring problems [11], and shown that it's performance is in line with the extensive evidence that demonstrates that the max-sum algorithm generates good approximate solutions when applied to cyclic graphs (in the context of approximate inference through 'loopy' belief propagation on Bayesian networks [4], iterative decoding of practical error correcting codes [5], and solving large scale K-SAT problems involving thousands of variables [6]). This algorithm effectively propagates information around the network such that it converge to a *neighborhood maximum*, rather than a simple local maximum [4].

Hence, having described a generic decentralised coordination algorithm, we then apply it within a specific problem concerning the self-organisation of sensors within a wireless network deployed within an urban environment to detect vehicle movements on a road network. Within this setting, energy management is a key challenge, and such sensors will typically 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 an indefinite lifetime [7]. However, since the sensing ranges of the sensors within the network will typically overlap with one another, the overall effectiveness of the sensor network depends not only on the sensors' individual choice of duty cycles, but also on the combined choice of neighbouring sensors whose sensing ranges overlap. With an ad-hoc sensor deployment, these interactions are not known prior to deployment, and thus, we describe how the sensors may self-organise by first learning the interactions between their neighbours (i.e. how much their neighbours' sensing

fields overlap with their own), and then coordinating their sense/sleep schedules (using the decentralised coordination algorithm described above) in order to address the system-wide performance goal of maximising the probability that a vehicle is detected. We show that by self-organising in this way, we can achieve a 30% reduction in the number of vehicles that the sensor network fails to detect (compared to an uncoordinated network), and this performance is shown to be close to that achieved by a benchmark centralised optimisation algorithm (simulated annealing).

The remainder of this paper is structured as following. In section 2 we present our factor graph representation and max-sum decentralised coordination algorithm. In section 3 we describe how it can be applied within self-organising sensors in our wide-area surveillance scenario amd we present our empirical evaluation. Finally, we conclude and discuss future work in section 4.

2 The Max-sum Approach to Coordination

The max-sum algorithm is a specific instance of a general message passing algorithm that exploits the *general distributive law* in order to decompose a complex calculation by factorising it (i.e. representing it as the sum or product of a number of simpler factors) [8]. Such algorithms are frequently used in fields such as information theory, artificial intelligence and statistics. In our case, they represent an ideal combination of the best features of optimal algorithms and approximate stochastic algorithms. They can make efficient use of constrained computational and communication resources, and yet are able to effectively represent and communicate complex utility relationships through the network and attain close to optimal solutions. In the following, we first provide a formal description of the generic coordination problem we address in this paper, and then detail the max-sum algorithm itself.

2.1 Problem Description

We initially consider the general case in which there are M sensors, and the state of each sensor may be described by a discrete variable x_m. Each sensor interacts locally with a number of other sensors such that the utility of an individual sensor, $U_m(\mathbf{x}_m)$, is dependent on its own state and the states of these other sensors (defined by the set $\mathbf{x}_m$). For example, in the wide area surveillance problem that we consider in this paper, the state of the sensor represents the sense/sleep schedule that it has adopted, the interacting sensors are those whose sensing areas overlap with its own, and the utility describes the probability of detecting an event within the sensor's sensing range. However, our approach at this stage is generic, and thus, we make no specific assumptions regarding the structure of the individual utility functions.

Within this setting, we wish to find the state of each sensor, $\mathbf{x}^*$, such that the sum of the individual sensors' utilities (commonly referred to as social welfare within the multi-agent systems literature) is maximised:

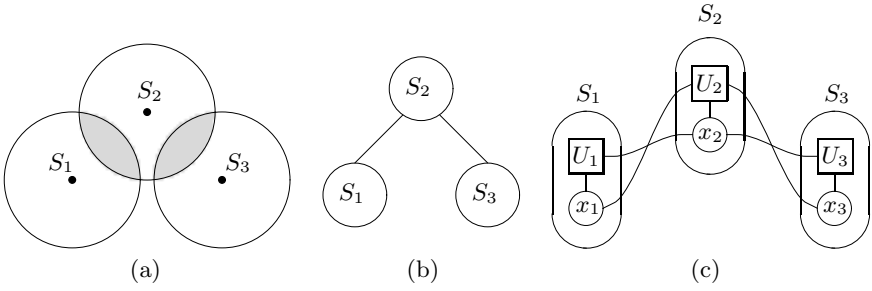

Fig. 1. Diagram showing (a) the position of three sensors in the environment whose sensing ranges overlap, and (b) the resulting factor graph with sensors decomposed into function and variable nodes

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

Furthermore, in order to enforce a truly decentralised solution, we assume that each sensor only has knowledge of, and can directly communicate with, the few neighbouring agents on whose state its own utility depends. In this way, the complexity of the calculation that the sensor performs depends only on the number of neighbours that it has (and not the total size of the network), and thus, we can achieve solutions that scale well.

2.2 Factor Graph Representation

In order to apply the max-sum algorithm, we represent the optimisation problem described in equation 1 above, as a bipartite factor graph. To this end, we decompose each sensor into a variable node that represents its state, and a function node that represents its utility. The function node of each sensor is connected to its own variable node (since its utility depends on its own state), and also to the variable nodes of other sensors whose states its utility depends on. For example, we show in figure 1a an example in which three sensors, $\{S_1, S_2, S_3\}$, interact with their immediate neighbours through the overlap of their sensing areas. In figure 1b we show the resulting constraint graph often used within the optimal algorithms, such as DPOP, discussed earlier. In figure 1c we show the resulting bipartite factor graph in which the sensors are decomposed into function nodes, $\{U_1, U_2, U_3\}$, and variable node, $\{x_1, x_2, x_3\}$. The overall function represented by this factor graph is given by $U = U_1(x_1, x_2) + U_2(x_1, x_2, x_3) + U_3(x_2, x_3)$.

2.3 Message Content of the Max-sum Algorithm

The max-sum algorithm operates directly on the factor graph representation described above. When this graph is cycle free, the algorithm is guaranteed to converge to the global optimal solution such that it finds the combination of states that maximises the sum of the sensors' utilities. When applied to cyclic

graphs (as is the case here), there is no guarantee of convergence but extensive empirical evidence demonstrates that such family of algorithms generate good approximate solutions [9,10].

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 nodes to variable nodes. These messages are defined as:

– **From variable to function**

$$q_{i\to j}(x_i) = \alpha_{ij} + \sum_{k\in\mathcal{M}_i\setminus j} r_{k\to i}(x_i) \tag{2}$$

where $\mathcal{M}_i$ is a vector of function indexes, indicating which function nodes are connected to variable node i, and α_{ij} is a scaler chosen such that $\sum_{x_i} q_{i\to j}(x_i) = 0$, in order to normalise the message and prevent them increasing endless in the cyclic graphs.

– **From function to variable**

$$r_{j\to i}(x_i) = \max_{\mathbf{x}_j\setminus i}\left[U_j(\mathbf{x}_j) + \sum_{k\in\mathcal{N}_j\setminus i} q_{k\to j}(x_k)\right] \tag{3}$$

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

Note that these message definitions are somewhat cyclic, and reflect the iterative way in which the messages are updated. The messages flowing into and out of the variable nodes within the factor graph are functions that represent the total utility of the network for each of the possible states of the variable. At any time during the propagation of these messages, sensor i is able to determine which state it should adopt such that the sum over all the sensors' utilities is maximised. This is done by locally calculating the function, $z_i(x_i)$, from the messages flowing into i's variable node:

$$z_i(x_i) = \sum_{j\in\mathcal{N}_i} r_{j\to i}(x_i) \tag{4}$$

and hence finding $\arg\max_{x_i} z_i(x_i)$.

Previous applications of the max-sum algorithm have applied it as an efficient iterative algorithm for centralised problems such as decoding error correcting codes [5]. Here, the factor graph is actually physically divided among the sensors within the network, and thus the computation of the system-wide global utility function is now carried out through a distributed computation involving message passing between sensors. Thus although the max-sum algorithm is approximating the solution to a global optimisation problem it involves only local communication and computation.

2.4 Convergence and Performance

The messages described above may be randomly initialised, and then updated whenever a sensor receives an updated message from a neighbouring sensor; there is no need for a strict ordering or synchronisation of the messages. In addition, the calculation of the marginal function shown in equation 4 can be performed at any time (using the most recent messages received), and thus, sensors have a continuously updated estimate of their optimum state. When the underlying factor graph contains cycles there is no guarantee that the max-sum algorithm will converge; nor that if it does converge it will find the optimal solution. However, extensive empirical evaluation on a number of benchmark coordination problems, including graph colouring, indicates that it does in fact produce better quality solutions than other state of the art approximate algorithms such as DSA, but at significantly lower computation and communication cost compared to complete algorithms such as DPOP [11].

2.5 Architecture

Note that the formulation of the problem as a factor graph does not actually limit where any of the computation is actually performed. At its most decentralised, each factor may reside on a single agent, and this agent may reside on a separate sensor. In this case, messages between factors and variables (and between variables and factors), represent messages exchanged between the sensors, across the communication network. However, it is also possible to use exactly the same factor graph representation as a computationally efficient means to perform the coordination even within a system where the elements are not actually distributed across different computational entities. Indeed, it is exactly this approach that is employed in the context of approximate inference through 'loopy' belief propagation on Bayesian networks [4], iterative decoding of practical error correcting codes [5], and when solving large scale K-SAT problems involving thousands of variables [6].

3 Wide Area Surveillance Problem

Having presented our max-sum coordination algorithm we now focus on its application within the self-organisation of a sensor network deployed for a wide area surveillance task. To this end, we consider a wide area surveillance problem based upon a simulation of an urban settings (using the Robocup Rescue Simulation Environment — see http://www.robocuprescue.org/). We assume that multiple wireless sensors are randomly deployed within the environment, and these sensors are tasked with detecting vehicles that travel along the roads. We assume that the sensors have no *a priori* knowledge of the road network, and do not know their own location within it. The sensors detect seismic or acoustic signals in order to indicate the binary presence, or absence, of vehicles within their sensing fields. We make no assumptions regarding the shape or range of these

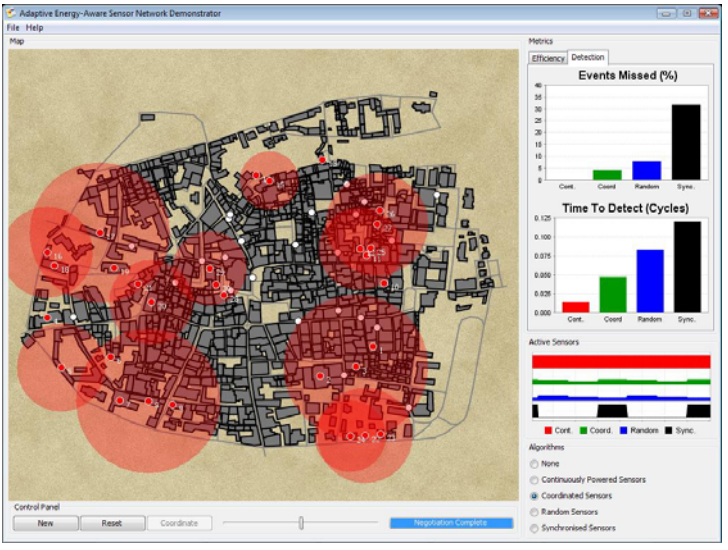

Fig. 2. Simulation of a wide area surveillance scenario (based on the Robocup Rescue Simulation Environment)

sensing fields (although for ease of simulating the setting, within our simulation we model these as circular fields with randomly assigned radii). Figure 2 shows this simulation environment in operation. The area sensed by active sensors is shown in red, and moving vehicles are shown as white markers on the roads.

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 field and communicate with other sensors)[1]. 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 indefinite 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 in this paper, we specifically consider schedules in which the sensor is in its sense state for one time slot, and in a sleep state for all $L - 1$

[1] Note that we assume that the energy consumed by activating the sensor is much more significant than that used in communication. This is generally true for sensors that require continuous signal processing such as the acoustic or seismic sensors considered here.

other time slots (i.e. $\sum_{k=0}^{L-1} s_k^i = 1$). This represents the simplest description of a power constrained sensing schedule, however, we note that the max-sum coordination algorithm that we have presented in the last section, can be applied for any discrete schedule.

3.1 The Coordination Problem

Figure 3 illustrates the coordination problem that results from this scenario. In this specific example, three sensors, $\{S_1, S_2, S_3\}$, are randomly deployed and exhibit overlapping sensing fields. In order to maintain energy neutral operation, each sensor can only actively sense for half of the time (i.e. $L = 2$), and thus, each sensor has a choice from two sensing schedules: either $\{1, 0\}$ or $\{0, 1\}$.

The system-wide goal is to maximise the probability that events are detected by the sensor network as a whole. This is achieved by ensuring that the area covered by the three sensors is actively sensed by at least one sensor at any time. However, with the sensing schedules available, it is clearly not possible to ensure that area $S_1 \cap S_2$, area $S_2 \cap S_3$ and area $S_1 \cap S_3$ are all sensed continually. Thus, the sensors must coordinate to ensure that the minimal area possible exhibits the minimal periods during which no sensor is actively sensing it. In this case, the optimal solution is the one shown where $\mathbf{s}_1 = \{0, 1\}$, $\mathbf{s}_2 = \{1, 0\}$ and $\mathbf{s}_3 = \{0, 1\}$. Note that this leads to areas $A_{\{1,2\}}$, $A_{\{2,3\}}$ and $A_{\{1,2,3\}}$ being sensed continually, and the smallest area, $A_{\{1,3\}}$, and of course the three non-overlapping areas, exhibiting intermittent sensing.

In a larger sensor deployment, each of these three sensors is also likely to overlap with other sensors. Thus, finding the appropriate sensing schedule of each sensors, such that probability of detecting an event is maximised, is a combinatorial optimisation problem. As such, this problem is similar to the graph colouring benchmark used in the evaluation of the max-sum algorithm described earlier [11]. However, an important difference is that in our sensor scheduling problem we can have interactions between multiple sensors (as is the case in the example shown in figure 3), rather than interaction between just pairs of sensors (as is the case in the standard graph colouring problem).

3.2 Applying the Max-sum Algorithm

To apply the max-sum coordination algorithm to this problem it is necessary to first decompose the system-wide goal that we face (that of maximising the probability that an event is detected) into individual sensor utility functions. As shown above, the utility of each sensor is determined by its own sense/sleep schedule, and by those of sensors whose sensing fields overlap with its own. Thus, we define $\mathbf{N}_i$ to be a set of indexes indicating which other sensors' sensing fields overlap with that of sensor i 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 sensor i and those sensors in $\mathbf{k}$. For example, with respect to figure 3, the area $A_{\{1,2\}}$ is the area that is sensed only by sensors 1 and 2. In a slight abuse of notation, we represent the entire sensing area of sensor S_1 as S_1, and thus, note that the area $A_{\{1,2\}}$ is different from

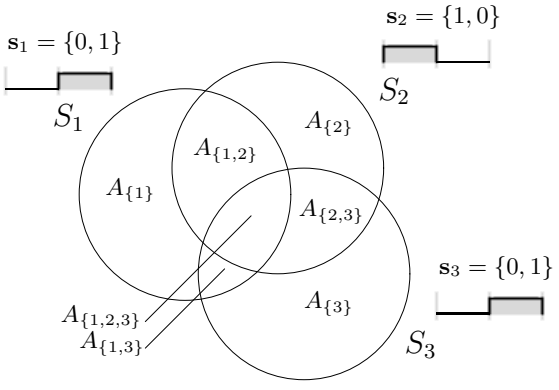

Fig. 3. Example coordination problem in which three sensors, $\{S_1, S_2, S_3\}$, have sensing fields that overlap

$S_1 \cap S_2$ because the area $S_1 \cap S_2$ would include also the sub area $S_1 \cap S_2 \cap S_3$. In general, we have:

$$A_{\{i\}\cup\mathbf{k}} = \bigcap_{j\in(\{i\}\cup\mathbf{k})} S_j \setminus \bigcup_{l\notin(\{i\}\cup\mathbf{k})} S_l$$

We define a function $G : 2^{\mathbf{X}} \rightarrow \mathcal{S}$ and $G(\mathbf{x}_{\{i\}\cup\mathbf{k}})$ is the combined sensing schedule of sensor i and those sensors in $\mathbf{k}$ (calculated through the logical 'OR' of each individual schedule). Now, assuming that events occur uniformly over the environment, then the utility of sensor i is given by:

$$U_i(\mathbf{x}_i) = \sum_{\mathbf{k}\subseteq\mathbf{N}_i} \frac{A_{\{i\}\cup\mathbf{k}}}{|\{i\}\cup\mathbf{k}|} \times P(\text{detection}|\lambda_d, G(\mathbf{x}_{\{i\}\cup\mathbf{k}})) \tag{5}$$

where $P(\text{detection}|\lambda_d, G(\mathbf{x}_{\{i\}\cup\mathbf{k}}))$ is the probability of detecting an event given the combined sensing schedules of the overlapping sensors and a parameter, λ_d, that describes the typical duration of an event. We model this duration as a Poisson process such that the probability of an event lasting time t is given by $\lambda_d e^{-\lambda_d t}$. Note, that we scale the area by the number of sensors 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. For example, the utility of sensor S_2 shown in figure 3, is calculated by considering the areas $A_{\{2\}}$, $A_{\{1,2\}}$, $A_{\{2,3\}}$ and $A_{\{1,2,3\}}$.

3.3 Learning the Mutual Interaction of Sensing Fields

The utility function presented in equation 5 assumes that the sensors are able to determine the area of overlap of their own and neighbouring sensors' sensing fields, and that they know the parameter λ_d that describes the detectable duration of events. In reality, sensors may have highly irregular and obscured sensing

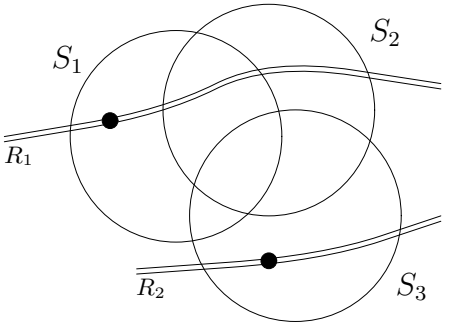

Fig. 4. Example showing the paths of two vehicles on roads, $\{R_1, R_2\}$, crossing the sensing fields of three overlapping sensors S_1, S_2 and S_3

areas, they may not be able to determine the exact position of themselves, let alone neighbouring sensors, and events may be known to be more likely to occur in some areas than others. Thus, we now relax these constraints, and describe how the sensors may learn these relationships in order to make a coordinated decision regarding their sense/sleep schedules. In more detail, we implement the following scheme:

Calibration Phase
We assume that after deployment, all sensors initially select the same sensing schedule, and thus, the sensors are all active and sense simultaneously[2]. At regular intervals during this phase sensors exchange information regarding the events that they have detected, and they keep track of (i) the number of events that they observe individually, O_i, and (ii) the number of events that are both detected by themselves and a subset of their neighbours, $O_{\{i\}\cup\mathbf{k}}$. The exact form that this exchange of information takes depends on the nature of the sensors used, and the events that they are detecting. Within our wide area surveillance scenario, we assume that sensors are able to time stamp the appearance and disappearance of vehicles within their sensing fields. Comparison of the time stamps of observations of other sensors then allow the sensor to identify whether vehicles are detected by multiple sensors as they cross its own sensing field.

For example, consider figure 4 in which the two vehicles crossing three overlapping sensing fields, and assume that sensor S_1 time stamps the appearance and disappearance of a vehicle at times 09:02:12 and 09:02:21 respectively, sensor S_2 time stamps the appearance and disappearance of a vehicle at times 09:02:15 and 09:02:24 respectively, and finally, sensor S_3 time stamps the appearance and disappearance of a vehicle at times 09:02:27 and 09:02:33 respectively. In this case, the intersection of the time stamps of sensors S_1 and S_2 lead these two

[2] Note that the performance of the network is somewhat degraded throughout this calibration phase since all the sensors select the same sensing schedule and are synchronised. We intend to address this within future work (see section 4).

sensors to conclude that $O_{\{1\}} = 1$, $O_{\{1,2\}} = 1$, $O_{\{2\}} = 1$, while the non-intersection of the time stamps of sensor S_3 leads it to conclude that $O_{\{3\}} = 1$. Note that in general, more complex techniques may be required to differentiate events when they occur concurrently. This will typically require some additional information such as the position of the event, or some recognisable characteristic of the event, and within the data fusion and tracking literature, this problem is commonly known as data or track association. Here, we assume that events are uniquely identified, since data association is not the focus of this paper. However, we note that this data association need not be error-free and the performance of the network degrades slowly if errors do occur.

Finally, the duration of the events is used to calculate an estimate of λ_d. This is easily done, since the maximum likelihood estimate of λ_d is simply the mean of the observed event durations.

Coordination Phase
The numbers of events observed in the calibration phase now acts as a proxy for the unknown areas of overlap between neighbouring sensors. Furthermore, it also captures the fact that events will not occur evenly over the entire area, but are restricted to certain areas (i.e. the roads in our case). Hence, the sensors now calculate their utility based on a modification of equation 5 given by:

$$U_i(\mathbf{x}_i) = \sum_{\mathbf{k} \subseteq \mathbf{N}_i} \frac{O_{\{i\} \cup \mathbf{k}}}{|\{i\} \cup \mathbf{k}|} \times P(\text{detection} | \lambda_d, G(\mathbf{x}_{\{i\} \cup \mathbf{k}})) \tag{6}$$

The sensors can now use the max-sum coordination algorithm presented earlier to coordinate their choice of sense/sleep schedule such that the utility of the overall sensor network is maximised, and hence, the probability of detection of a vehicle traveling within the area covered by the sensor network is maximised.

Operational Phase
Finally, the operational phase proceeds as before, sensors simply follow the sense/sleep schedule determined in the previous coordination phase. If during this phase a sensor fails, then the coordination algorithm above may simply be re-run to coordinate over the smaller sensor network. Likewise, should the position of sensors change, or new sensors be added, both the calibration phase and the coordination phase can be re-run to coordinate over the new environment in which the sensors find themselves. In section 4 we shall describe our future work developing a more principled approach that allows for continuous self-adaption of the sensor network as the state of the environment, or the sensors themselves, changes over time.

To validate this approach we now perform an empirical evaluation within our simulation environment.

3.4 Empirical Evaluation

To this end, we simulated the above three phases using random deployments of the sensors whose sensing ranges are assumed to be circular discs with radius

drawn uniformly between $0.05d$ and $0.15d$ (where d is the maximum dimension of the area in which the sensors are deployed). During the calibration phase we simulated the movement of 500 vehicles between random start and end points, and the sensors exchanged observations with one another regarding their observations during this time. During the coordination phase, the sensors use the max-sum algorithm over a fixed number of cycles, in order to coordinate their sensing schedules. Finally, during the operational phase the sensors use the sensing schedules determined in the negotiation phase, and we again simulate the movement of 500 vehicles between random start and end points.

We measure the operational effectiveness of the sensor network by calculating the percentage of vehicles that are missed by the sensor network (i.e. vehicles that move between their start and end point without ever being within the sensing field of an actively sensing sensor) and for those vehicles that are detected, we measure the time taken for the first detection (i.e. the time at which the network first becomes aware of the presence of the vehicle after it leaves its start point). We repeat the experiments 100 times for three different length sensing schedules ($L = 2$, 3 and 4) and we investigate three different ranges of sensor number such that the effective number of sensors (given by N/L) remained constant. In this way, each deployment had the same effective sensing capability. We compare results for four different coordination mechanisms:

1. **Randomly Coordinated Sensors**
   As before, the choice of each sensors' sense/sleep schedule is made randomly by each individual sensor with no coordination.

2. **DSA Coordinated Sensors**
   Using the results of the calibration phase, the sensors use the DSA algorithm (as discussed earlier) to coordinate their sense/sleep schedules. The probability that any agent actually updates its preferred state is 0.4 (determined through empirical optimisation).

3. **Max-sum Coordinated Sensors**
   Using the results of the calibration phase, the sensors use the max-sum algorithm to coordinate their sense/sleep schedules.

4. **Simulated Annealing Coordinated Sensors**
   We use an offline centralised optimisation algorithm to calculate an upper bound on the performance that we can expect from our decentralised max-sum approach. This solution cannot be used in practice to coordinate the sense/sleep schedules of the real sensors since it is centralised and assumes full knowledge of the topology of the network, however, it provides an upper bound on the performance of a coordinated solution.

The results of these experiments are shown in figures 5 and 6, where the error bars represent the standard error of the mean in the repeated experiments. In more detail, figure 5 shows the percentage of vehicles that fail to be detected by

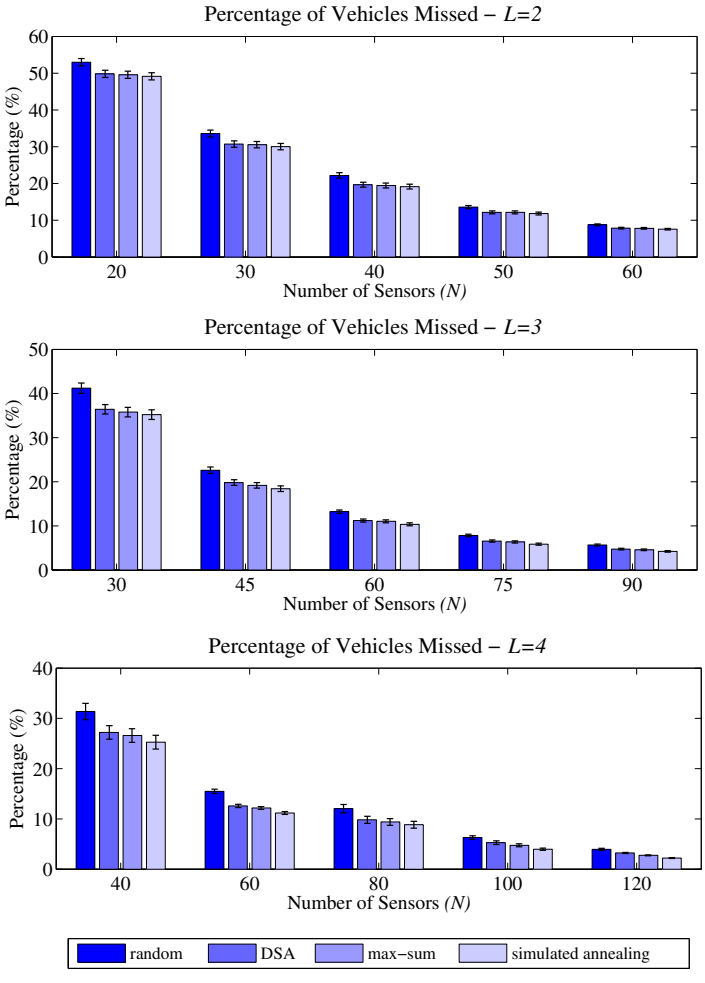

Fig. 5. Comparison of simulation results reporting the percentage of missed vehicles, for a sensor network using random, DSA, max-sum, and centralised simulated annealing coordination algorithms plotted against the number of deployed sensors

the sensor network; our main metric for the performance of the network. Figure 6 shows the time that it took the sensor network to first detect each vehicle; a metric that we do not actively seek to minimise. Note that in all cases, the randomly coordinated sensor network performs the worst (failing to detect more vehicles and taking a longer time to detect them), and that the centralised simulated annealing approach provides the best solutions. In each case, the max-sum approach out performs the DSA algorithm, and does so without the need to empirically tune a parameter. The difference between the algorithms increases as both the number of sensors within the network and the length of

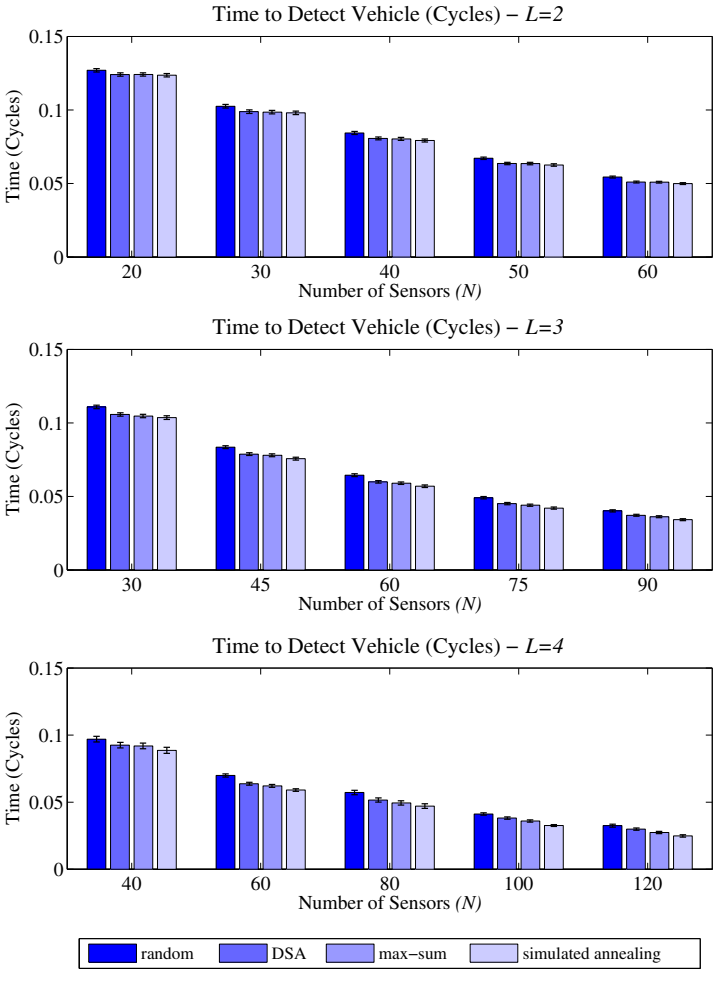

Fig. 6. Comparison of simulation results reporting the mean time to first detect a vehicle, for a sensor network using random, DSA, max-sum, and centralised simulated annealing coordination algorithms plotted against the number of deployed sensors

sensing schedules increase. This trend is expected as the combinatorial coordination problem becomes harder as both these factors increase.

Table 1 shows the results for both of these metrics for the case when $L = 4$ and $N = 120$. In this case, by using our approach, we achieve a 30% reduction in the number of missed vehicles (compared to the uncoordinated network), and this performance is shown to be close to that achieved by the benchmark centralised optimisation algorithm (simulated annealing) and significantly better than DSA (a state of the art decentralised approach).

Table 1. Comparison of percentage of vehicles missed and time to detect vehicles for each coordination algorithm when $L = 4$ and $N = 120$

Coordination Algorithm	Percentage of Vehicles Missed (%)	Time to Detect Vehicle (Cycles)
random	4.0±[0.4]	0.033±[0.002]
DSA	3.2±[0.2]	0.030±[0.002]
max-sum	2.7±[0.2]	0.028±[0.002]
simulated annealing	2.2±[0.2]	0.025±[0.002]

4 Conclusions

In this paper, we have considered the self-organisation of sensors within a network deployed for wide area surveillance. We have presented a decentralised coordination algorithm based upon the max-sum algorithm and demonstrated how self-organisation can be achieved within a setting where sensors are deployed with no *a priori* information regarding their local environment. We showed the sensors can learn how their actions interact with those of neighbouring sensors, and then use the max-sum algorithm to coordinate their sense/sleep schedules in order to maximise the effectiveness of the sensor network as a whole. In a software simulation we showed that this approach yields significant improvements in performance over the case of random coordination, and closely approaches that of a benchmark centralised optimisation algorithm.

Our future work consists of extending these results in order to relax the requirement of a separate calibration phase prior to the negotiation phase. The synchronised schedules of the sensors during the calibration phase corresponds to a period of poor system-wide performance that is offset by improved system-wide performance during the operational phase. However, it is also possible to learn about the occurrence of events, and hence the overlap of sensors' sensing fields, during this operational phase. Thus, we would like to investigate online algorithms that can trade-off between exploratory behaviour (synchronising with neighbouring sensors to learn about the occurrence of events), and exploitative behaviour (using relationships already learnt to coordinate the sensors). Principled Bayesian heuristics exist for performing such trade-offs [12], and applying these approaches within this setting would remove the requirement for three distinct phases. Rather, the sensors would continuously self-organise and self-adapt, changing sense/sleep schedules continuously to trade-off between exploration and exploitation. Such an approach would also naturally apply within dynamic settings where sensors' utilities may change at any time, sensors may fail, or additional sensors may be deployed. The max-sum coordination algorithm used in this paper already supports this continual behaviour since utility messages can be communicated, and sensors can estimate their optimal state, at anytime, and thus, it would appear to be a solid base on which to develop this more advanced self-organising behaviour.

References

1. Rogers, A., Corkill, D.D., Jennings, N.R.: Agent technologies for sensor networks. IEEE Intelligent Systems 24(2), 13–17 (2009)
2. Petcu, A., Faltings, B.: DPOP: A scalable method for multiagent constraint optimization. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence, pp. 266–271 (2005)
3. Fitzpatrick, S., Meetrens, L.: Distributed Coordination through Anarchic Optimization. In: Distributed Sensor Networks A Multiagent Perspective, pp. 257–293. Kluwer Academic, Dordrecht (2003)
4. Weiss, Y., Freeman, W.T.: On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. IEEE Transactions on Information Theory 47(2), 723–735 (2001)
5. MacKay, D.: Good error-correcting codes based on very sparse matrices. IEEE Transactions on Information Theory 45(2), 399–431 (1999)
6. Mezard, M., Parisi, G., Zecchina, R.: Analytic and algorithmic solution of random satisfiability problems. Science 297(5582), 812–815 (2002)
7. Kansal, A., Hsu, J., Zahedi, S., Srivastava, M.B.: Power management in energy harvesting sensor networks. ACM Transactions on Embedded Computing Systems 6(4) (2007)
8. Aji, S., McEliece, R.: The generalized distributive law. IEEE Transactions on Information Theory 46(2), 325–343 (2000)
9. Kschischang, F.R., Frey, B.J., Loeliger, H.A.: Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory 42(2), 498–519 (2001)
10. MacKay, D.J.C.: Information Theory, Inference, and Learning Algorithms. Cambridge University Press, Cambridge (2003)
11. Farinelli, A., Rogers, A., Petcu, A., Jennings, N.R.: Decentralised coordination of low-power embedded devices using the max-sum algorithm. In: Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems, pp. 639–646 (2008)
12. Dearden, R., Friedman, N., Andre, D.: Model based Bayesian Exploration. In: Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, pp. 150–159 (1999)