A Decentralised Coordination Algorithm for Minimising Conflict and Maximising Coverage in Sensor Networks

Ruben Stranders, Alex Rogers and Nicholas R. Jennings
School of Electronics and Computer Science
Southampton, SO17 1BJ, UK.
{rs06r,acr,nrj}@ecs.soton.ac.uk

ABSTRACT

In large wireless sensor networks, the problem of assigning radio frequencies to sensing agents such that no two connected sensors are assigned the same value (and will thus interfere with one another) is a major challenge. To tackle this problem, we develop a novel decentralised coordination algorithm that activates only a subset of the deployed agents, subject to the connectivity graph of this subset being provably 3-colourable in linear time, hence allowing the use of a simple decentralised graph colouring algorithm. Crucially, while doing this, our algorithm maximises the sensing coverage achieved by the selected sensing agents, which is given by an arbitrary non-decreasing submodular set function. We empirically evaluate our algorithm by benchmarking it against a centralised greedy algorithm and an optimal one, and show that the selected sensing agents manage to achieve 90% of the coverage provided by the optimal algorithm, and 85% of the coverage provided by activating all sensors. Moreover, we use a simple decentralised graph colouring algorithm to show the frequency assignment problem is easy in the resulting graphs; in all considered problem instances, this algorithm managed to find a colouring in less than 5 iterations on average. We then show how the algorithm can be used in dynamic settings, in which sensors can fail or new sensors can be deployed. In this setting, our algorithm provides 250% more coverage over time compared to activating all available sensors simultaneously.

Categories and Subject Descriptors

I.2.9 [**Computing Methodologies**]: Artificial Intelligence—*Distributed Artificial Intelligence*; I.2.9 [**Computing Methodologies**]: Artificial Intelligence—*Robotics: Sensors*

General Terms

Algorithms, Experimentation, Theory

Keywords

Sensor Networks, Distributed Problem Solving, Graph Colouring

1. INTRODUCTION

Recently, the use of wireless micro-sensor networks has generated a significant amount of interest in areas such as climate change research [12], weather and tidal surge prediction [11], and monitoring intelligent buildings [7]. These networks consist of cheap sensors with very limited computational capabilities; potentially, they can be deployed by scattering them from airplanes or ground vehicles. Taken to the extreme, these sensors could even become the size of a grain of sand or even dust, in which case they are also referred to as *smartdust* [17].

Crucially, because of their limitations constrained computational resources, these sensors need simple and robust algorithms for controlling various aspects of the network. Now, since centralised control of such aspects is often not possible, and would introduce a single point of failure, the use of agent-based technologies have been advocated to solve these problems in a decentralised fashion [15]. In particular, such approaches include decentralised solutions for packet routing [12], efficient prediction of missing data [11], and assigning radio frequencies (or, equivalently, time slots for the Time Division Multiple Access protocol [1]) to sensors such that the number of required retransmissions due to interference is minimised (and data throughput is maximised), which is the problem we focus on in this paper.

Now, several aspects of this problem have already been studied in the literature. In particular, the frequency assignment problem, which is often cast as a multi-agent graph colouring problem, has been extensively addressed by the development of various message-passing algorithms [5, 6, 13, 1]. However, these algorithms either produce approximate solutions [5, 6, 1], or require exponential computation and communication [13]. These properties push these algorithms beyond the limited computational and communicational abilities of micro-sensors, or require a large number of frequencies to optimally colour the graph, which reduces the bandwidth of the network [1].

Therefore, in this paper, we propose a different approach to this problem: instead of solving the graph colouring problem in the original sensor network, we develop a novel decentralised coordination algorithm that deactivates certain sensors within the network, such that the resulting connectivity graph is more easily colourable. More specifically, the algorithm constructs a *triangle-free* graph, which does not contain cliques of size greater than 2. This is appealing because it is a known result that these types of graphs are 3-colourable [16] in linear time [3]. Equally important, this bounds the number of required frequencies, which can be very large for the original sensor network.

However, this poses a new question: how to select those sensors which should be deactivated to ensure that the resulting sensor network has maximum coverage? In the literature, the problem of maximising sensing coverage has been studied in both the presence [7] and the absence [4] of a centralised controller. In these settings, sensing coverage is often modeled as a submodular set function, a versatile mathematical abstraction that intuitively captures the diminishing returns of adding an extra sensor to a sensor network.

Cite as: A Decentralised Coordination Algorithm for Minimising Conflict and Maximising Coverage in Sensor Networks, Ruben Stranders, Alex Rogers and Nicholas R. Jennings, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. XXX-XXX.

For example, the probability of event detection with arbitrary sensing areas, entropy reduction [9] and mutual information [7].

We show that the problem of selecting a triangle-free sensor network that maximises coverage is NP-hard (based on theoretical results from [10]). Therefore, we develop an approximate algorithm; it allows sensors to coordinate in a fully decentralised fashion to build a triangle-free algorithm with high sensing coverage.

In more detail, the contributions of this paper are:

1. We derive a novel decentralised algorithm that activates a subset of the available sensors so as to maximise sensing coverage given by an arbitrary submodular set function, subject to the connectivity graph being triangle-free.

2. We develop a centralised greedy algorithm based on the notion of submodular independence systems, and derive a theoretical lower bound of $1/7$ on the approximation ratio of the algorithm, for any submodular function. This algorithm acts as a benchmark for the decentralised algorithm in the empirical evaluation.

3. We develop dynamic counterparts for these two algorithms that are capable of dealing with failing sensors and new sensors, while ensuring the triangle-free property of the graphs.

4. We empirically evaluate our algorithm and show that it provides 90% of the coverage provided by an optimal algorithm, and 85% of the coverage provided by activating all sensors (without the restriction on the graph). Moreover, we show that the frequency assignment problem on the resulting sensor network can be solved by a simple and standard decentralised graph colouring algorithm. Finally, in the dynamic setting, we show that our algorithm provides 250% more coverage over time compared to activating all available sensors simultaneously.

The remainder of this paper is organised as follows. In Section 2 we define the problem. In Section 3 we present a centralised and a decentralised algorithm. In Section 4 we extend these algorithms to operate continually to replace failing sensors. In 5 we empirically evaluate this set of algorithms and demonstrate their effectiveness. Finally, we conclude in Section 6.

2. PROBLEM DESCRIPTION

Let $\mathcal{S} = \{S_1, \ldots, S_M\}$ denote a set of M sensors deployed on the $\mathbb{R}^2$ plane. The Cartesian coordinates are given by a vector $\mathbf{x_i} = (x_i, y_i)$. Let $d(\mathbf{x_i}, \mathbf{x_j})$ denote the Euclidean distance between S_i and S_j. Each sensor S_i has a radio disk with radius r_i within which other sensors can receive their transmissions. Consequently sensor S_j can receive S_i's transmissions iff S_j is contained within S_i's radio disk: $d(\mathbf{x_i}, \mathbf{x_j}) \leq r_i$. Each sensor S_i has control over its transmission radius r_i, which it set anywhere between 0 and r_{max}, which is the maximum transmission radius for all sensors. Given this model, we can construct a *connectivity graph* that models the communication network that exists among the sensors:

Definition 1. A sensor *connectivity graph* $C[\mathcal{S}]$ of a set of sensors $\mathcal{S}$ is an directed graph $C[\mathcal{S}] = (\mathcal{S}, E)$ in which E contains a pair of sensors (S_i, S_j) if S_j can receive S_i's transmissions.

Now, to ensure transmissions between two sensors are not compromised by interference from other sensors, we wish to allocate frequencies to each sensor such that no two sensors with overlapping radio disks are allocated the same transmission frequency. However, note that the connectivity graph only models *direct* collisions, which are those that occur between S_i and S_j if they are contained within each other's radio disks; it does not model the possibility of *indirect* collisions [1], which occur when two sensors S_i and S_j are not contained within each others radio disks, but there exists a sensor S_k that is contained in both (see Figure 1 for an example). When indirect collisions occur, sensor S_k will receive garbled transmissions from S_i and S_j. Therefore, instead of using $C[\mathcal{S}]$, we consider the *collision graph* $C^2[\mathcal{S}]$:

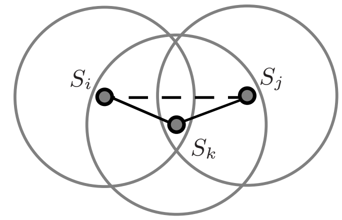


Figure 1: The connectivity and collision graph of an example sensor network. Possible direct and indirect collisions are represented by solid and dashed edges respectively.

Definition 2. The *collision graph* of sensors $\mathcal{S}$ is the square of $C[\mathcal{S}]$, denoted by $C^2[\mathcal{S}]$. This graph contains an edge (S_i, S_j) if there exists a path between S_i and S_j in $C[\mathcal{S}]$ of at most two edges.

By effectively connecting neighbours of neighbours in the connectivity graph, the collision graph models the possibility of direct as well as indirect collisions. Thus, solving the frequency allocation problem is equivalent to colouring $C^2[\mathcal{S}]$, which is a known NP-complete problem. Now, to bound the number of required colours needed to colour the graph (also referred as the chromatic number), which can be very large on an arbitrary sensor network, and also in the interest of keeping the sensors as simple and robust as possible, we proceed in two steps. First, we wish to find a set of sensors whose connectivity graph $C[\mathcal{S}]$ is easily colourable. More specifically, by this, we mean that the graph is *triangle-free*. A triangle-free graph is a graph that does not contain any cycles of length 3, or, equivalently, whose maximum clique size is 2. A 3-colouring of a triangle-free graph is guaranteed to exist [16], and can be computed in linear time [3]. This colouring avoids any *direct* collisions. In the second step we attempt to avoid any *indirect* collisions by considering the denser *collision graph* of this triangle-free connectivity graph. Simple graph theory shows that this graph is guaranteed to be K_7 minor-free,[1] based on the triangle-free property of the connectivity graph. By exploiting this property, and applying the famous Hadwiger conjecture [8] we know that the obtained collision graph is 6-colourable.[2] Thus, the maximum number of colours needed to colour the collision graph of a triangle-free connectivity graph is 6. In Section 5, we show that this can be achieved by a simple decentralised graph colouring algorithm.

Besides ensuring reliable communication between the sensors, we also wish to maximise the sensing quality that the sensor network provides. For reasons discussed in the introduction, in this paper, sensing quality is given by a submodular set function:

Definition 3. A set function $f : 2^E \to \mathbb{R}$ defined over a finite set E is called *submodular* if for $A \subseteq B \subseteq E$ and $e \in E$, $f(A + e) - F(A) \geq F(B + e) - F(B)$.

In more detail, sensing quality achieved by a subset of $\mathcal{S}$ is given by a non-decreasing submodular function $f : 2^{\mathcal{S}} \to \mathbb{R}^+$. Intu-

[1] A K_7 minor-free graph does not contain the complete graph K_7 as a subgraph, i.e. it contains no cliques larger than 6.

[2] The Hadwiger conjecture states that any K_k minor-free graph is $(k-1)$-colourable. It has been proven for $k \leq 6$ [14], but in this paper, we assume that the conjecture holds for $k = 7$ as well.

itively, function f defines the diminishing returns of adding an extra sensor to an existing sensor network.

Thus, the problem we address in this paper is to find a set $\mathcal{S}' \subseteq \mathcal{S}$ that maximises $f(\mathcal{S}')$ subject to $C[\mathcal{S}']$ being triangle-free. These sensors $\mathcal{S}'$ will then form the new sensor network, by deactivating sensors $\mathcal{S} \setminus \mathcal{S}'$.

Additionally, we can also consider a dynamic version of this problem, by taking into account that sensors can fail. One major cause of sensor failure is battery depletion. In this paper, we assume that radio transmission accounts for the majority of the energy consumption of the sensor (and thus do not consider the energy required for sensing). In more detail, sensors have an initial battery capacity b_i, which reduces over time as a result of their transmission power as follows: $\Delta b_i = -r_i^2 \cdot \Delta t$.

In the upcoming sections, we show how, by activating a subset of the sensor network such that the connectivity graph is triangle-free, we can use greedy algorithms to compute sensor deployments $\mathcal{S}'$ with good sensing quality.

3. SENSOR SELECTION ALGORITHMS

In this section, we first present a centralised greedy algorithm with theoretical bounds on the solution. This algorithm will act as a benchmark for the decentralised algorithm that we develop in Section 3.2. We then show that both algorithms activate a subset of the deployed sensors whose connectivity graph is not necessarily connected. As a result, sensors will not always be able to communicate their measurements to a base station. Therefore, in Section 3.3, we develop a decentralised algorithm that attempts to reconnect the various components of the graph by incrementally increasing their communication range.

Because of the correspondence between the results for the triangle-free connectivity graph and its collision graph, in the remainder of the paper, we will consider the connectivity graph only.

3.1 A Centralised Greedy Algorithm

Before introducing a decentralised algorithm to this problem, we will first develop a centralised greedy algorithm based on the notion of independence systems from combinatorial optimisation.

Definition 4. An *independence system* is a pair $(E, \mathscr{I})$, where E is a finite set of elements, and $\mathscr{I}$ is a collection of subsets of E such that if $A \in \mathscr{I}$ and $B \subseteq A$, then $B \in \mathscr{I}$. Sets in $\mathscr{I}$ are said to be *independent*.

Clearly, the set $\mathscr{I}_{\triangle\text{-free}}$ of subsets of $\mathcal{S}$ whose connectivity graph is triangle-free form an independence system, since every induced subgraph of a triangle-free graph is triangle-free. Now, since not every subset is equal in terms of sensing quality, we augment these independence systems with the submodular function f that measures sensing quality:

Definition 5. A *submodular independence system* is an independence system together with a non-decreasing submodular set function f.

Unfortunately, for a submodular independence system, finding a set $I^* \in \mathscr{I}$ such that $I^* = \arg\max_{I \in \mathscr{I}} f(I)$ is a NP-hard problem [10]. Therefore, under the assumption that P $\neq$ NP, there does not exist a polynomial time algorithm for computing I^*. As a result, to obtain solutions that scale well with the size of the sensor network, we have to resort to approximation. One of the simplest approximation algorithms is the greedy algorithm (Algorithm 1), that builds a solution without backtracking by iteratively adding those elements that most improve the solution (with respect to f), while simultaneously satisfying an independence constraint.[3]

Algorithm 1 The greedy algorithm for a submodular independence system $((E, \mathscr{I}), f)$

```
I := ∅
while E ≠ ∅ do
    e* := arg max_{e∈E} f(I + e) − f(I)
    E := E − e*
    if I + e* ∈ 𝒥 then
        I := I + e*
    end if
end while
return I
```

The greedy algorithm computes a *maximal independent* set, which is an independent set I such that by adding any $e \in E \setminus I$, it becomes dependent. In other words, no sensor can be added to the sensor network without introducing a triangle.

Now, while the greedy algorithm is simple, it has its drawbacks; the main one being that it can perform arbitrarily badly, as illustrated by the following example:

Example 1. Let $E = \{A, B_1, \ldots, B_M\}$, $I = \{\{B_1, \ldots, B_M\}, \{A\}\}$, and $f(\{A\}) = n$, $f(\{B_i\}) = n - \epsilon$, $f(\{B_1, \ldots, B_M\} = (n - \epsilon)M$. The result of the greedy algorithm is $I = A$ after a single iteration. When $M \to \infty$ and $\epsilon \to 0$, the approximation ratio for Algorithm 1, i.e. $f(I)/f(I^*)$, approaches 0. In other words, for arbitrary independence systems, the greedy solution can be arbitrarily far away from the optimal solution.

Fortunately, many independence systems exhibit additional structure that can be exploited to obtain a lower bound on the approximation ratio for Algorithm 1. The notion of p-*independence* is one of these [2]:

Definition 6. An independence system $(E, \mathscr{I})$ is called p-*independent* if for all $A \in \mathscr{I}$ and $e \in E$ there exists a set $B \subseteq A$ such that such that $|B| \leq p$ and $A \setminus B + e \in \mathscr{I}$.[4]

The following is a result in combinatorics that proves a lower bound on the approximation ratio of the greedy algorithm:

THEOREM 1 ([2, 10]). *Algorithm 1 yields a $1/(1+p)$-approximation to maximising a non-decreasing submodular set function subject to a p-independence constraint.*

Thus, to obtain a lower bound on the greedy algorithm for the sensor coverage problem, we need to determine p for $(\mathcal{S}, \mathscr{I}_{\triangle\text{-free}})$. In order to do so, we need to restrict the problem defined in Section 2 slightly. This restriction involves limiting the radius of the radio disks to a constant R for every sensor.[5] A set of sensors with a fixed radio range R is denoted as $\mathcal{S}_R$, and the connectivity graph obtained is called a unit disk graph. This construct allows us to prove the following theorem:

THEOREM 2. *System $(\mathcal{S}_R, \mathscr{I}_{\triangle\text{-free}})$ is 6-independent.*

PROOF. Simple geometry shows that the maximum degree of a triangle-free unit disk graph no larger than 11. Let A be an valid solution (i.e. $A \in \mathscr{I}$). Now, $deg(e) \leq 11$ in $A + e$, otherwise $A \notin \mathscr{I}$. When $deg(e) = 11$, $A + e$ contains 11 triangles. To break

[3] Note that in this algorithm, the independence system $\mathscr{I}$ need not be explicitly given. Typically, an oracle in the form of an algorithm or indicator function $\mathbf{1}_{\mathscr{I}}(\mathcal{S}) = true \Leftrightarrow \mathcal{S} \in \mathscr{I}$ suffices.

[4] When $p = 1$, a p-independence system is called a *matroid*, which is a well-known structure in combinatorics.

[5] We will drop this restriction again in our empirical evaluation, and show that this has no detrimental effect on the algorithm's performance.

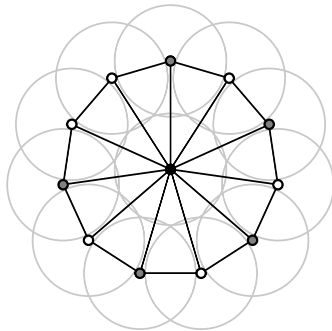


Figure 2: Visual representation of the proof of Theorem 2. See text for explanation.

each of these, we remove $p = \lceil 11/2 \rceil = 6$ vertices from A. Let B denote this set of vertices. Then $A \setminus B + e \in \mathscr{I}$ with $|B| \leq 6$, as required. $\square$

See Figure 2 for a visual representation of this proof. In this figure, e is the black vertex, B is represented by the white vertices and A is represented by the white and gray vertices. Radio disks are represented by gray circles. (For ease of exposition, these radio disks have been scaled by 50%. As a result, links exists within this unit disk graph when the scaled disks overlap.)

As a result of Theorem 2, the greedy algorithm is guaranteed to produce a solution I such that $f(I)/f(I^*) \geq 1/7$ for system $(\mathcal{S}_R, \mathscr{I}_{\triangle\text{-free}})$. However, we do not know whether or not this lower bound is tight. For example, note that in the worst case, greedy yields a $6/11$ approximation on the construction used in the proofs.[6] Moreover, our empirical evaluation (see Section 5) obtained approximation ratios no less than 75%, even without the requirement that $r_i = R$ for all i.

3.2 A Decentralised Greedy Algorithm

Thus far we have assumed the existence of a centralised controller that has perfect knowledge of function f, set E and has a way of determining whether the resulting graph is triangle-free. However, when a centralised controller is absent, sensors do not have access to the global knowledge required to execute the greedy algorithm in a purely decentralised fashion. While it might be possible to construct a decentralised algorithm that shares all required information, this would require excessive communication between sensors. Therefore, in this section, we take a different approach and develop an effective approximate decentralised algorithm that requires very limited local communication. In more detail, this algorithm allows sensors to construct a triangle-free network by inspecting their neighbourhood.[7] That is, if the neighbourhood of all sensing agents is triangle-free, the connectivity graph is triangle free.

Now, the principle behind the algorithm is that no more than two sensors within a clique can be activated without creating a triangle. The key challenge is then to find which two sensors should be activated to maximise sensing quality. Obviously, since solving this problem optimally in a central fashion is NP-hard, solving it optimally in a decentralised fashion is at least as hard. Therefore, we (again) resort to approximation in the form of a *greedy* decentralised algorithm. Using this algorithm, sensors are able to

[6]This occurs when $f(e) = n$, and for all $a \in A$, $f(a) = n - \epsilon$. The greedy selection of e in the first iteration blocks the addition of 5 of 11 elements in A. Thus, the worst case approximation ratio is $\lim_{\epsilon \to 0} f(A \setminus B + e)/f(A) = 6/11$.

[7]The *neighbourhood* of a vertex is the subgraph induced by the vertex and its adjacent vertices.

Algorithm 2 The Distributed Greedy Algorithm for S_i

1: $State \leftarrow$ BASIC
2: $\mathcal{S}_i^B \leftarrow adj(S_i)$
3: Broadcast $\langle \mathbf{i}, \mathcal{S}_i^B \rangle$
4: Receive $\langle \mathbf{j}, \mathcal{S}_j^B) \rangle$ for all $S_j \in adj(S_i)$
5: **while** $State =$ BASIC **do**
6:   On random activation
7:   $\mathcal{S}_i^B \leftarrow \{S \mid S \in \mathcal{S}_i^B \wedge State(S) \neq \text{DOMINATED}\}$
8:   **if** $\exists j : \mathcal{S}_i^B \cap \mathcal{S}_j^B \neq \varnothing$ **then**
9:     Randomly select $S_k \in \mathcal{S}_i^B \cap \mathcal{S}_j^B$
10:     $f_{jk} \leftarrow f(\{S_j, S_k\})$
11:     **if** $f_{jk} \geq f(\{S_i, S_j\})$ and $f_{jk} \geq f(\{S_i, S_k\})$ **then**
12:       $State \leftarrow$ DOMINATED
13:     **end if**
14:   **else**
15:     $State \leftarrow$ DOMINATING
16:   **end if**
17:   Send $\langle \mathbf{i}, State \rangle$ to all $S_j \in adj(S_i)$
18: **end while**

coordinate with their neighbours to identify the sensors that maximise coverage within that clique. In more detail, when running this algorithm, each sensor S_i continually checks whether a pair of sensors (S_j, S_k) exist within the same clique, such that the coverage provided by (S_j, S_k) is greater than the coverage provided by both (S_i, S_j) and (S_i, S_k). If this is discovered to be the case, S_i is said to be DOMINATED. In all other cases, S_i is said to be DOMINATING. In the former case, activating the sensor would result in suboptimal sensing quality, and the sensor would turn itself off. Similarly, in the latter it is better to activate the sensor.

In more detail, Algorithm 2 captures the necessary steps to determine the status of a sensor. Before starting the main **while** loop, neighbours are discovered by means of message passing (lines 3 and 4). Then, in lines 7 and 8, the sensor attempts to find a non-DOMINATED neighbour that in turn has a non-DOMINATED neighbour in common with itself (i.e. a triangle). If no such neighbour can be found, the sensor's best strategy is to turn itself on (line 15). If, however, such a neighbour *does* exist, at least one of these three sensors needs to turn off in order to ensure that the graph is triangle-free. Therefore, in line 11, the algorithm checks whether turning itself on is a dominated strategy. If this is the case, the sensor sets its state to DOMINATED, notifies its neighbours of its updated status, and turns itself off (line 12).

A sensor is capable of detecting termination of this algorithm by inspecting the states of its neighbours: if all neighbours are either DOMINATED or DOMINATING, the algorithm has terminated. Note that termination of this algorithm is guaranteed: when the number of iterations approaches infinity, a DOMINATED sensor will select S_k in Line 9, such that (S_j, S_k) with probability 1, and deactivate itself. All DOMINATING sensors will remain in the BASIC state, until all DOMINATED sensors have deactivated themselves. At this point, DOMINATING sensors will no longer be able to find a triangle (Line 8), and thus detect their DOMINATING state (Line 15).

3.3 The Reconnection Phase

At this point we have constructed two greedy algorithms that compute a triangle-free subgraph of $C[\mathcal{S}]$ with high sensing quality. However, note that these induced subgraphs are not necessarily strongly connected,[8] since they only satisfy the requirement that they are triangle-free (or cycle-free). We therefore add a second

[8]A graph is *strongly connected* if there exists a path from every vertex to every vertex. For a sensor connectivity graph, this means that every sensor is capable of communicating with all other sensors via multi-hop routing. In the remainder of the paper, when we use the term *connected* we mean *strongly connected*.

phase to these algorithms, that attempts to reconnect different components of the computed subgraph. This algorithm is not always capable of fully reconnecting the graph, since the maximum radio transmission range r_{max} is sometimes insufficient. However, as we will show in Section 5, even when it is not completely successful, it manages to increase the size of the maximum connected component significantly.

Now, to attempt to reconnect the graph, sensors incrementally boost their radio signals to connect with sensors that were previously unreachable. Naturally, it is undesirable for sensors to set their radio strength to the maximum setting, for two reasons. Firstly, it will drain their battery quickly (with a rate of r^2_{max}), thereby reducing the operation time of the sensor network as a whole. Secondly, by increasing the radius of its radio disk, a sensor might introduce additional edges that create a triangle, which was exactly what we set out to avoid. So, in order to connect to additional sensors whilst maintaining a triangle-free graph, each sensor individually executes Algorithm 3 which relies on local computation and communication only.

Algorithm 3 The Reconnection Algorithm

1: **repeat**
2: Broadcast $\langle \mathbf{i}, adj(S_i) \rangle$
3: Receive $\langle \mathbf{j}, adj(S_j) \rangle$ for all $S_j \in adj(S_i)$.
4: $r_i \leftarrow c \cdot r_i$
5: **until** $adj(S_i) \cap adj(S_j) \neq \varnothing$ or $r_i > r_{max}$
6: $r_i \leftarrow r_i / \sqrt{c}$

It is easy to see that this algorithm preserves the triangle-free property of the first connectivity graph. This is achieved by continually checking whether a neighbouring sensor shares a common neighbour. If it is discovered that this is the case (Line 5), the graph contains a triangle, at which point both neighbours reduce their radio range in order to break it (Line 6).

To illustrate the techniques we have developed in this Section, Figure 3 shows the output of the centralised algorithm for an example sensor deployment with $M = 100$ sensors. In this example, sensor quality is represented by different sized sensing disks. Notice how phase 1 selects a subgraph that consists of 8 components. Phase 2 reconnects these components effectively whilst ensuring that the resulting connectivity graph remains triangle-free. A particularly attractive feature of the selected sensors, is that their connectivity graph is colourable with 3 colours, and their collision graph with 6 colours (see Section 2), while the original connectivity graph (in this particular case) needed 23 colours (so its collision graph would probably require $\gg 23$ colours).

4. DEALING WITH DYNAMISM

In the previous section, we discussed algorithms that perform a one-off optimisation procedure to activate a subset of the sensors that provide high sensing quality. In this section, we consider a more dynamic setting, in which deployed sensors can fail and new sensors can be deployed. Now, as the example in Figure 3 and the experimental results in Section 5 will show, the number of sensors needed by both the centralised and decentralised algorithms is fairly small compared to the number of deployed sensors. Thus, the remaining sensors that are not selected by either algorithm can be used to replace failed sensors. In this section, we will therefore develop dynamic counterparts of Algorithm 1 and Algorithm 2, that continuously monitor the sensor network and select replacements for sensors that stop functioning. These dynamic counterparts are obtained as follows:

Centralised: The key property of the centralised greedy algorithm is that it selects the sensors that most improve the already constructed solution. So, once a sensor fails, Algorithm 1 is run again. However, instead of initialising I to the empty set in line 1, I is initialised to the already computed subset, minus the failing sensors. Furthermore, E is initialised to E minus all active and failed sensors. The algorithm will then proceed to iteratively add new sensors (if possible). Should new sensors be deployed, these are simply added to E, and the algorithm will run as before.

Decentralised: Instead of completely turning off DOMINATED sensors, these sensors keep monitoring communication in their neighbourhood. Once a neighbouring sensor fails (which can be detected by a prolonged interval of communication silence), it resets its state to BASIC, and runs Algorithm 2 again. Active sensors (i.e. those with a DOMINATING state) need not re-run the algorithm. Should new sensors be deployed, they will be treated as DOMINATED sensors.

5. EMPIRICAL EVALUATION

In this section, we will evaluate the algorithms developed in the previous sections in a large scale sensor deployment scenario.

In the first part of the empirical evaluation, we measure the performance of the non-dynamic versions of the centralised and decentralised greedy algorithms. In the second part, we subject the dynamic versions of these algorithms to empirical evaluation. First, however, we describe the experimental setup common to both.

5.1 Experimental Setup

To empirically evaluate the algorithms, we consider a scenario in which M sensors have been randomly deployed in a unit square, and are tasked with event detection. The area in which sensor S_i can detect an event is a disk with radius s_i, which is drawn from the interval $[0.05, 0.2]$ with uniform likelihood. The radius r_i within which sensor S_i can receive and send transmissions is uniformly drawn from the interval $[0.5R, R]$, where R controls the range of r_i, and is one of the parameters we vary in our experiments. Moreover, the maximum radio transmission range is $r_{max} = 1.2R$. Events occur randomly within the unit square with uniform probability. The sensing quality $f(\mathcal{S}')$ achieved by a subset $\mathcal{S}' \subseteq \mathcal{S}$ is the expected number of detected events, i.e. those that fall within the sensing disk of at least one sensor.[9]

More formally, let $\triangledown(S_i)$ denote the sensing area of sensor S_i, i.e. $\triangledown : \mathcal{S} \rightarrow 2^{\mathbb{R}^2}$. Moreover, let $\mu(\cdot)$ denote the measure[10] of an area, i.e. $\mu : 2^{\mathbb{R}^2} \rightarrow \mathbb{R}$. Now, function f is defined as:

$$f(\mathcal{S}) = \mu \left(\bigcup_{S \in \mathcal{S}} \triangledown(S) \right) \tag{1}$$

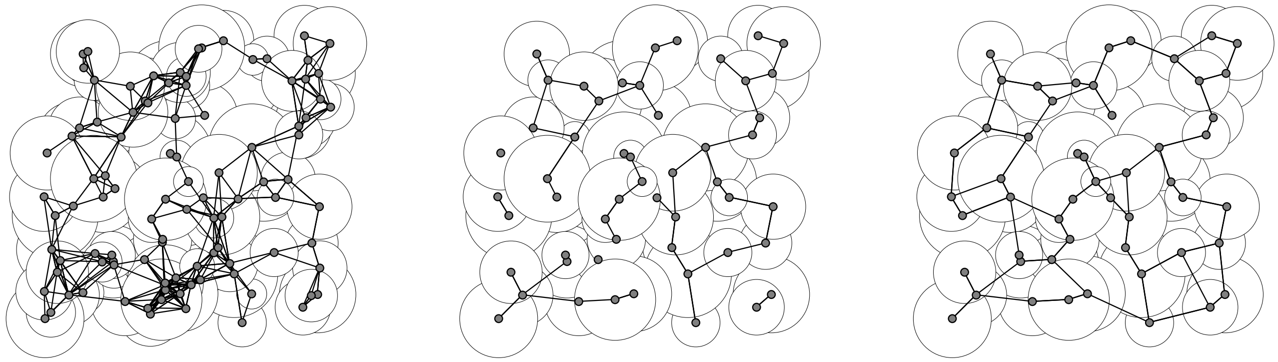
Clearly, f is a non-decreasing submodular function, since adding a sensor S_i to a deployment $\mathcal{S}$ increases total coverage less than adding S_i to $\mathcal{S}' \subset \mathcal{S}$.

5.2 Evaluation of the Greedy Algorithms

For the first set of experiments, we applied the centralised and decentralised greedy algorithms of Section 3 on a simulated sensor deployment with $M = 300$. The sensing range for each sensor was uniformly drawn from $[0.05, 0.2]$, while the radio range was

[9] We have also considered scenarios with different parameters, for which our algorithms performed equally well. However, we found that these specific parameters lead to particularly challenging problem instances.

[10] To avoid confusion, we use the term 'area' to describe two dimensional shapes, and 'measure' to denote the extent (or size) of an area.

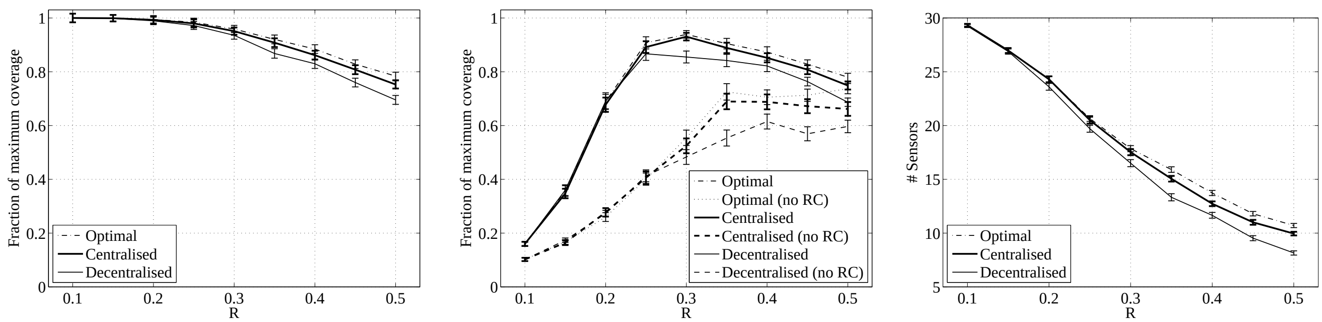


(a) The original sensor deployment of 100 sensors.

(b) Sensors selected by the centralised greedy algorithm.

(c) The connectivity graph after the reconnection phase.

Figure 3: Example execution of the algorithm. The circles represent the sensing areas of the sensor. An edge between two sensors indicates communication between them is possible.

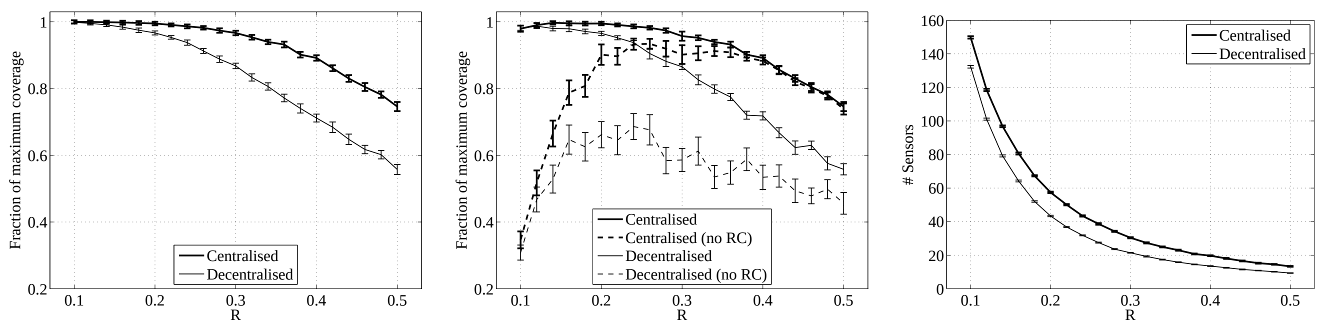


(a) Achieved fraction of maximum possible sensing coverage.

(b) Achieved fraction of maximum possible sensing coverage by largest connected component.

(c) Number of active sensors.

Figure 4: Results for the static algorithms ($M = 30$). Error bars indicate the error of the mean.



(a) Achieved fraction of maximum possible sensing coverage.

(b) Achieved fraction of maximum possible sensing coverage by largest connected component.

(c) Number of active sensors.

Figure 5: Results for the static algorithms ($M = 300$). Error bars indicate the error of the mean.

uniformly drawn from $[0.5R, 1.0R]$. During the experiments, we varied R between 0.1 and 0.5, to determine the effect of different levels of density in the connectivity graph $C[\mathcal{S}]$.

We benchmarked the algorithms against an optimal algorithm that computes a triangle-free subgraph with optimal coverage. This algorithm uses branch and bound and exploits the structure of submodular functions to improve computational efficiency. Despite these computational efficiency improvements, however, such an optimal approach does not scale beyond ≈ 30 sensors.[11] Because of this, we performed two batches of experiments. In the first, we used 30 sensors and evaluated the centralised, decentralised and optimal algorithms, and in the second batch we applied the centralised and decentralised algorithms on a deployment of 300 sensors.

We measured the sensing coverage of the selected sensors computed by both centralised and the decentralised algorithms as a frac-

[11]In more detail, on many problem instances, the optimal algorithm took >2 hours, while both greedy algorithms always terminated in less than 5 seconds on a standard desktop computer.

tion of the sensing coverage of all sensors. Moreover, in order to determine the effectiveness of the reconnection algorithm, we also measured the coverage achieved by the largest connected component of the graph. This metric captures the trade-off between the graph connectedness and sensing quality. Finally, we measured the number of selected sensors.

The results of the first batch are summarised in Figure 4. Figure 4(a) shows the sensing quality as a fraction of the sensing quality achieved by all M sensors. This plot clearly shows that the difference between the optimal solution and the solution computed by both greedy algorithms is less than 10% in the most constrained case (i.e. $R = 0.5$). This is a clear indication that both greedy algorithms compute very good approximations, without the need for exhaustively searching the solution space. Figure 4(b) shows the sensing quality achieved by the largest component. In this figure, the postfix 'no RC' indicates that the reconnection algorithm from Section 3.3 was not used. This figure demonstrates the effectiveness of the reconnection algorithm; it manages to connect a sufficient number of components to almost double the sensing quality of the largest component of the graph. Finally, Figure 4(c) shows that the optimal algorithm manages to select a small number of extra sensors compared to both greedy algorithms. As expected both greedy algorithms are less successful in satisfying the independence constraints while maximizing sensor coverage. However, this effect is only marginal, since the optimal algorithm selects just 10% more sensors than the decentralised greedy algorithm.

The results of the second batch are shown in Figure 5. Overall, the same features as before can be observed here. However, Figure 5(a) shows that the achieved coverage of the decentralised algorithm drops below 60% of the maximum achievable coverage for $R = 0.5$. The same—albeit less strong—effect can be observed for the centralised greedy algorithm. However, note that for this level of radio range, the sensors cover around a quarter of the entire area. As a result, the connectivity graph of the original sensor network is very dense, and by limiting the solution to triangle free graphs the problem is very constrained. Figure 5(b) again demonstrates the effectiveness of the reconnection algorithm, but also that between $R = 0.1$ and $R = 0.3$ both algorithms provide at least 85% of the maximum possible sensor coverage, while needing approximately half (for $R = 0.1$) to a tenth for ($R = 0.3$) of the available sensors.

Finally, to corroborate the theoretical result that the resulting connectivity and collision graphs are easily 3 and 6-colourable respectively (see Section 2), we used a simple and standard algorithm to colour the graphs in a decentralised fashion. This algorithm is an ϵ_n-greedy algorithm, i.e. with probability $1 - \epsilon_n$ it selects the colour that minimises the number of mono-chromatic edges, while with probability ϵ_n it picks a random colour. Furthermore, probability ϵ_n decreases with each iteration of the algorithm as $\epsilon_n \leftarrow \frac{n-1}{n}\epsilon_n$, where n is the iteration number. Given this context, Figure 6 shows that colouring the resulting graphs in a decentralised fashion is indeed trivial: for 300 sensors, the algorithm needs 5 iterations on average to correctly colour both types of graphs. Moreover, this simple algorithm managed to find a colouring in all 5000 considered problem instances.

5.3 Evaluation of the Dynamic Algorithms

For the second experiment, we evaluated the dynamic greedy algorithms. To do this, we simulated randomly deployed sensor networks as before. However, we now also consider sensor failures. In our simulations, sensors get deactivated as they completely deplete their battery. Initially, every sensor S_i has a battery capacity b_i of 1 unit. Recall from Section 2 that the battery depletion rate is modeled as $\Delta b_i = -r_i^2 \cdot \Delta t$. Each time a sensor fails, we employ the algorithms developed in Section 4 to attempt to replace it with sensors that were not selected for the initial deployment. We

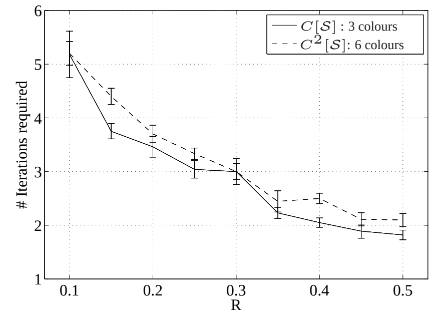


Figure 6: Iterations required by the greedy graph colouring algorithm ($M = 300$).

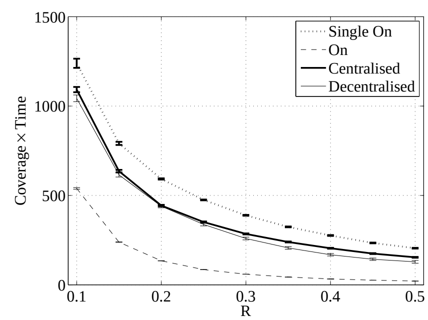
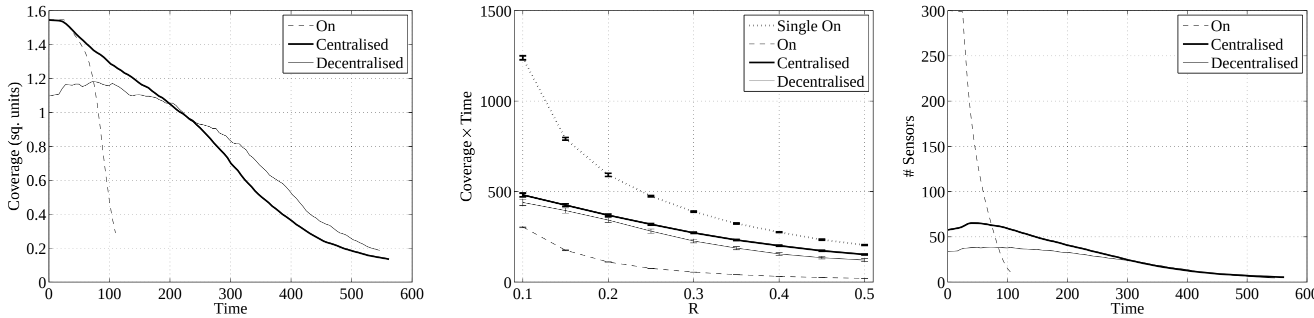


Figure 8: Total coverage over time.

benchmarked our algorithms against a naïve strategy (referred to as 'On') in which all sensors are activated upon deployment.

The results are shown in Figure 7. The plots in Figure 7(a) and 7(b) show the coverage over time achieved by all active sensors and the largest component respectively. Clearly, the sensing quality provided by the 'On' strategy rapidly decreases, since all sensors are activated, and the sensing areas overlap, causing redundancy. Compared to 'On' the triangle-free and decentralised algorithms perform notably better. Moreover, whereas the decentralised algorithm is outperformed by the centralised one for the initial sensor deployment (cf. Figures 5(a) and 5(b)), the decentralised algorithm starts outperforming its centralised counterpart after $t \approx 250$. The explanation for this is found in Figure 7(c) that shows the number of active sensors over time: the decentralised algorithm requires less sensors for the initial deployment, and therefore has more sensors available to replace failed ones.

Finally, we recorded the total sensor coverage provided over time for several radio ranges R. Sensor coverage over time is defined as the area of the region below the graphs shown in Figures 7(a) and 7(b). For this experiment, we added an additional benchmark strategy that activates only a single sensor at a time (referred to as 'Single On'). The performance of this strategy acts as an upper bound on the total coverage over time that can be achieved, since no two sensors redundantly cover the space. Figures 8 and 9 show the results. These figures confirm that 'On' is outperformed by both greedy algorithms for several values of R, and by around 250% for $R = .2$. Moreover, by comparing the performance of our algorithms to that of 'Single On', we see that these algorithms manage quite effectively to minimise redundant coverage, since 'Single On' has no redundant coverage by its very nature. The most important conclusion we can draw from these experiments, however, is that the decentralised algorithm achieves at least 80% of the sensing quality of the centralised greedy algorithm (92% for $R = .2$), while only requiring local communication and computation.



(a) Coverage over time (b) Coverage over time (largest component) (c) Active sensors over time

Figure 7: Results for the dynamic algorithms ($M = 300$, $R = 0.2$).

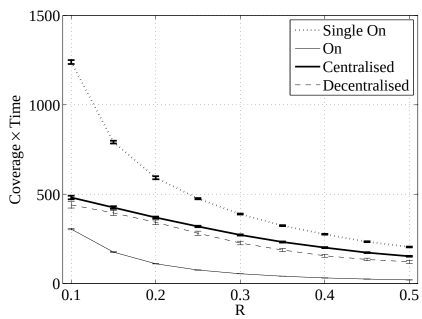


Figure 9: Total coverage over time of largest component.

6. CONCLUSIONS

In this paper, we developed a novel decentralised algorithm that activates a subset of a wireless sensor network, such that the connectivity graph of the resulting sensors is triangle-free, while at the same time maximising sensing coverage. Moreover, we also developed a centralised greedy algorithm based on the notion of submodular independence systems, for which we derived a theoretical lower bound of $1/7$ on the approximation ratio of the algorithm. We then proceeded to consider dynamic settings, in which sensors can fail, or new sensors can be added to the existing deployment, and extended both our algorithm and the centralised greedy algorithm to operate in such settings. We empirically evaluated our algorithm by benchmarking it against the centralised algorithm and an optimal one. We showed that the selected sensors manage to achieve 90% of the coverage provided by the optimal algorithm, and 85% of the coverage provided by activating all sensors. Equally important, we showed that the frequency assignment problem in the resulting sensor network can be solved by a simple decentralised graph colouring algorithm. Finally, in the dynamic setting, our algorithm provides 250% more coverage over time compared to activating all available sensors simultaneously.

Our future work in this area is to extend the applicability of these algorithm to passive mobile sensors. These are sensors that are moved by forces beyond their control, such as wind or water. Since the connectivity graph will be subject to constant change, the computed subgraph of the sensor network might have to be periodically revised. Furthermore, we would like to investigate the use of decentralised scheduling algorithms (e.g. [4]) to reduce redundant coverage by overlapping sensing areas, which would improve the lifetime of the network even further.

Acknowledgments This research was undertaken as part of the ALADDIN (Autonomous Learning Agents for Decentralised Data and Information Systems) Project and is jointly funded by a BAE Systems and EPSRC (Engineering and Physical Research Council) strategic partnership (EP/C 548051/1).

7. REFERENCES

[1] K. L. Bryan, T. Ren, D. L., T. Henry, and V. Fay-Wolfe. Towards optimal TDMA frame size in wireless sensor networks. Technical report, University of Rhode Island, 2007.

[2] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a submodular set function subject to a matroid constraint (extended abstract). In *IPCO '07: Proceedings of the 12th international conference on Integer Programming and Combinatorial Optimization*, pages 182–196, 2007.

[3] Z. Dvořák, K. Kawarabayashi, and R. Thomas. Three-coloring triangle-free planar graphs in linear time. In *SODA '09: Proc. of the Nineteenth Annual ACM -SIAM Symposium on Discrete Algorithms*, pages 1176–1182, 2009.

[4] A. Farinelli, A. Rogers, and N. Jennings. Maximising sensor network efficiency through agent-based coordination of sense/sleep schedules. In *Workshop on Energy in Wireless Sensor Networks in conjuction with DCOSS*, 2008.

[5] 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 Multi-Agent Systems*, pages 639–646, Estoril, Portugal, 2008.

[6] S. Fitzpatrick and L. Meertens. Distributed coordination through anarchic optimization. In *Distributed Sensor Networks*, chapter 11, pages 257–295. Kluwer Academic Publishers, 2003.

[7] C. Guestrin, A. Krause, and A. P. Singh. Near-optimal sensor placements in gaussian processes. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 265–272, 2005.

[8] H. Hadwiger. über eine klassifikation der streckenkomplexe. *Verteljahrsschr. Naturforsch, Ges. Zurich*, 88:133–142, 1943.

[9] C. W. Ko, J. Lee, and M. Queyranne. An exact algorithm for maximum entropy sampling. *Operations Research*, 43(4):684–691, 1995.

[10] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions—II. *Mathematical Programming Studies*, 8:73–87, 1978.

[11] M. A. Osborne, A. Rogers, S. D. Ramchurn, S. J. Roberts, and N. R. Jennings. Towards real-time information processing of sensor network data using computationally efficient multi-output gaussian processes. In *Proc. of the 7th International Conference on Information Processing in Sensor Networks*, pages 109–120, 2008.

[12] P. Padhy, R. K. Dash, K. Martinez, and N. R. Jennings. A utility-based sensing and communication model for a glacial sensor network. In *Proc. of the 5th International Conference on Autonomous Agents and Multi-Agent Systems*, pages 1353–1360, New York, NY, USA, 2006. ACM Press.

[13] A. Petcu and B. Faltings. DPOP: A scalable method for multiagent constraint optimization. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2005)*, pages 266–271, Edinburgh, Scotland, 2005.

[14] N. Robertson, P. Seymour, and R. Thomas. Hadwiger's conjecture for K_6-free graphs. *Combinatorica*, 13:279–361, 1993.

[15] A. Rogers, D. D. Corkill, and N. R. Jennings. Agent technologies for sensor networks. *IEEE Intelligent Systems*, 24(2):13–17, 2009.

[16] C. Thomassen. Grötzsch's 3-color theorem and its counterparts for the torus and the projective plane. *Journal of Combinatorial Theory, Series B*, 62(2):268–279, 1994.

[17] B. Warneke, M. Last, B. Liebowitz, and K. S. J. Pister. Smart dust: Communicating with a cubic-millimeter computer. *Computer*, 34(1):44–51, 2001.