Coordinated Intelligent Power Management and the Heterogeneous Sensing Coverage Problem

Richard Tynan, Conor Muldoon, Gregory O Hare and Michael O Grady

CLARITY: The Centre for Sensor Web Technologies, School of Computer Science and Informatics, University College Dublin, Belfield, Dublin 4, Ireland
Email: richard.tynan@ucd.ie

One of the most important factors to be considered when developing an application for a Wireless Sensor Network (WSN) is its power consumption. Intelligent Power Management (IPM) for a WSN is crucial in maximising the operational longevity. An established regime for achieving this is through the opportunistic hibernation of redundant nodes. Redundancy, however, has various definitions within the field of WSNs and indeed multiple protocols, each operating using a different definition, coexist on the same node. In this paper, we advocate the use of a MAS as an appropriate mechanism by which different stake-holders, each desiring to hibernate a node in order to conserve power, can collaborate. The problem of node hibernation for the heterogeneous sensing coverage areas is introduced and the manner by which it can be solved using ADOPT, an algorithm for distributed constraint optimisation, is described. We illustrate that the node hibernation strategy discussed here is more useful than the traditional stack-based approach and motivate our discussion using intelligent power management as an exemplar.

Keywords: Coverage, Connectivity, Hibernation, Intelligent Power Management, Distributed Constraint Optimisation

1. INTRODUCTION

The intelligent power management of a WSN provides an interesting application area for multi-agent systems. One of the primary metrics by which to measure the performance of a WSN is longevity. Due to the inherent remote operation and potential scale of a deployed WSN, replenishing node power supplies is both costly and time consuming, typically to the point of being impractical. While some approaches have harnessed natural solar [1] and vibrational energy [2] from the environment to allow nodes to operate indefinitely, these techniques lead to intermittent and unreliable node activity and are not considered in this paper. One strategy to prolong the life-span of the network is to deploy additional redundant nodes from the outset, which remain in a low power dormant state until such time as they are required [3]. The trend in reducing both node size and cost [4] means that the environmental and budget impact of the extra nodes will be small enough to permit such an over deployment. Once redundant nodes are deployed, a protocol is required to maintain the specified density by hibernating and activating appropriate nodes. Two such protocols are Coverage Configuration Protocol (CCP) [3] or Optimal Geographical Density Control (OGDC) [5].

Numerous definitions of redundancy exist within the field of WSNs, for example a node can be considered redundant if the other active nodes will remain connected in its absence [6]. Alternatively, a sensing redundancy definition would consider a node for hibernation if its sensing area is covered by one or more other nodes [7, 8]. A naive implementation of the later approach would hibernate the node without considering the effect of this on the connectivity of the remaining nodes. Hibernating a critical node could disconnect a large, outlying region of the network from the base station, resulting in a blind spot. It is this interdependency between different aspects of power management that require the cooperation of different entities hosted on a node. For some techniques, such as CCP or OGDC, when sensing coverage is achieved connectivity is assured automatically under certain conditions.

In this article, the heterogeneous sensing coverage and hibernation problem is introduced. The heterogeneous coverage problem is concerned with situations in which there are multiple coverage areas associated with each node in a WSN and whereby nodes sense a number of different phenomena, such as light, temperature, and sound. This article discusses how the problem is addressed using ADOPT [9], an algorithm developed for Distributed Constraint OPtimisation Problems (DCOP) [10, 11]. With distributed constraint optimisation algorithms, the goal is to answer the following question: "How do a set of agents optimise over a set of constraints such that a solution is found with some degree of global quality?". The constraint optimisation problem in general is known to be NP hard. Thus, it is necessary to use approximate approaches for large problem instances. ADOPT provides one such approach, however, the current reference implementation has been developed as a simulator and for standard Java. The implementation discussed in this article has been developed for Java Micro Edition (JME) and has been tested on Sun SPOT motes. One of the main advantages of ADOPT is that it provides a theoretical or analytical bounded error on the approximation. This bounded error can be increased or decreased by varying a threshold value. It provides a principled approach in the trade-off between solution accuracy and resource consumption.

There are several reasons why an application developer would like to optimally manage heterogeneous sensing coverage areas. For instance, they may wish the sensor network deployment to be capable of supporting multiple applications rather than a single application as has traditionally been the case with WSNs. In the future, there will be increasing pressure for multiple application support if WSNs are to be deployed in real world applications whereby companies will demand the maximum return from their investment. Another alternative scenario is that a single application will require different sensing capabilities to monitor the environment in question.

In the next section, background information, which is pertinent to the remainder of this paper, is detailed. This includes, additional power management issues, such as components to hibernate and decision deadlines, along with existing work to bring MAS technology to WSNs. Following this, our initial stack-based approach to solving this issue is presented. We then provide information on the mechanism by which cross-layer hibernation can be achieved, after which the proposed alternative based on a MAS located on the node is given. We then look at how the limited resources of a typical WSN node must constrain the reasoning capabilities of an agent operating in this environment. Architectures for local reasoning, such as the BDI model, are primarily concerned with the individual behaviour of agents. In Section 7, we discuss how DCOP algorithms can be used to facilitate collaborative behaviour. Following on from this, we discuss how the ADOPT algorithm is used to solve the heterogeneous sensing coverage and hibernation problem. Experimental results are then provided, which motivates the use of a MAS to allow the individual power managers to operate autonomously. Section 10 provides some concluding remarks.

2. BACKGROUND

Power management in a WSN takes many forms; a number of them are detailed in this section. Additionally, information on some of the existing agent frameworks for WSNs is presented as the basis for our proposed solution requires the use of a fine-grained MAS on an individual node rather than a one-agent-per-node strategy as has been adopted previously.

2.1. Intelligent Power Management

There are a number of different forms IPM for a WSN, from hibernating the entire node to switching off various components of the node e.g., the sensor or portions of memory [12]. Power management is also related to tuning various components, such as transmission power or sampling frequency, without hibernating the node [13, 14]. Typically, any decision to alter the state of the node affects all applications residing on it, and as such co-operation is vital to ensure the correct operation of all stake-holders while the management of the shared resources of the node is taking place.

Node hibernation is a particularly strict form of power conservation, since all operation of the node ceases, however, it yields significant increases in node longevity [3]. With this in mind, the decision to hibernate must be reached in a comprehensive and inclusive manner. Further complications exist when considering how long to hibernate a node for and also how often to re-evaluate each nodes redundancy. Redundancy is typically evaluated by nodes broadcasting a HELLO packet containing attributes, such as its location for example. Neighbouring nodes receive this information and manage a neighbour table, which is consulted to decide on redundancy. Inevitably some of these packets get lost and on each evaluation of redundancy the state of the network can change dramatically, and may not accurately reflect the topology produced in the presence of a perfectly reliable channel.

Some services on a WSN, such as routing [15], struggle to cope with a dynamic network topology and, as such, they favour the more stable topology resulting from a larger evaluation and sleep period [16]. On the other hand, some redundancy identification techniques favour shorter times so that the nodes can adapt to changes in the sensed data or failed nodes [17]. This poses a conflict when it comes to scheduling the services. If each service operates at the same time and on the same evaluation frequency, sub optimal performance of some or all layers will result.

If each service operates independently, then a further problem of aggregating redundancy decisions exists. For example, at time T the sensing redundancy component decides that this node is redundant. At time T + 10s the connectivity maintenance protocol also decides that the node is redundant. Should the previous redundancy decision persist or should it be discarded? Even with homogenous timing, protocols differ as to the duration of time before the decision is available and so the problem remains. The independent timing issue along with the required co-operation between entities, we believe, are some of the strongest motivating factors for using autonomous agents to deliver IPM on a WSN node.

2.2. Agents for WSNs

Agents have been deployed previously on WSNs, primarily to process the raw data in an intelligent fashion with a view to reducing transmissions - the single biggest factor in determining the longevity of the network [13, 18]. Other applications have used agents for the routing of packets around the network [19]. Agents have also been used in lighting control for intelligent energy conservation [20]. The application of agent technologies to the field of WSNs has been increasing steadily over the last few years, in particular, the underlying agent support frameworks for WSNs.

Agilla [21] is a middleware platform for deploying mobile agents, essentially mobile code. The agent architecture is tailored toward the computational constraints typical of WSN nodes. It allows for multiple agents to exist on a single sensing node, and provides methods for the reliable movement of agents between nodes. Sensing platforms provide context to an agent through tuples (a set of predefined descriptors about the node) in a tuple-space. The tuple-space also serves as the communication forum between agents on a node. Additionally, agents register their interests in particular events by inserting a template tuple into the tuple-space. Matching events are reported to the agent without the need for continuous polling.

Mate [22] allows WSN programs to be written in TinyScript, a scripting language which is compiled into executable bytecodes for an application-specific virtual machine. Allowing the virtual machine to be application-specific means that the programs for it can be clear and concise and thus less prone to failure, but this approach reduces compatibility of agents across different platforms. The bytecodes are less like mobile agents, but rather are like intentional viruses. Once a single instance of a bytecode program is introduced to the network, it automatically spreads by controlled flooding until all nodes of the network have a copy of the program. It is intended that only one program should operate on the network at once, and so this limits the flexibility of Mate as a basis for an agent system.

The ongoing improvement in WSN node technology has lead to the emergence of JavaME enabled devices such as the iMote2 [23] and SunSpot running the SQUAWK JVM [24]. Such developments pave the way for the porting of existing Java based agent environments to the field of WSNs. For example, Agent Factory [25] already has a version, Agent Factory Micro Edition (AFME) [26], capable of running on the Sun SPOT WSN mote and the Stargate platform, a device almost equivalent to the iMote2 sensor node. This facilitates the execution for multiple reflective agents in a WSN, along with agent mobility across heterogeneous devices within the network.

It should be noted, however, that there are significant differences between devices, such as the Sun SPOT, and other Java based motes, such as the Sentilla JCreate; for example, the amount of computational and power resources available. Devices such as the Sentilla have similar specification to a T-mote Sky, whereas the Sun SPOT would be closer to a mobile phone in terms computational capabilities.

3. STACK BASED APPROACH

Our initial solution at implementing an IPM system is depicted in Figure 1. The protocol stack uses interfaces to pass information up and down through the layers to provide the communication between nodes. The layers are ordered in this way to (1) enable the sensing coverage protocol to provide an appropriate density to the application and (2) enable the connectivity maintenance protocol to provide a connected topology over which to route packets. We now examine each layer in turn, starting at the lowest.

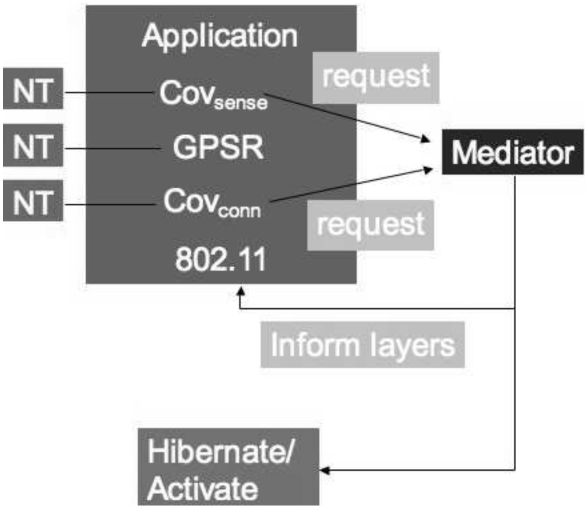

FIGURE 1. Protocol stack to deliver IPM. Both the connectivity coverage maintenance scheme and the sensing coverage protocol must agree on a nodes redundancy for hibernation to take place. NT represents the layers neighbour table.

When multiple nodes wish to communicate, they cannot do so at the same time due to interference on the channel, so a MAC layer is required in order to mediate the use of the channel and to retransmit failed packets. As such the first layer on the WSN device for this system architecture will be the MAC layer with direct control over the transceiver. For WSNs, numerous approaches to this have been developed, including B-MAC [27]. We, however, have opted for the 802.11 MAC layer implementation that comes with JSim [28]. Built on top of the MAC layer is the connectivity maintenance component. This ensures connectivity is maintained as nodes are hibernated according to the sensing requirement.

Next, a routing protocol is required to forward packets through multiple hops to their destination. We have selected the in built Greedy Perimeter Stateless Routing (GPSR) protocol [15] that comes as an optional package with J-Sim. Above this layer the sensing redundancy protocol operates, ensuring that an appropriate density of nodes is presented to the application. In our work, we have implemented two such techniques, CCP [3] and an Interpolation based Redundancy Identification and Sensor Hibernation (IRISH) protocol [17]. The application itself deals with messages from the base station and other nodes and provides the function of the WSN. Orthogonal to this stack is the required mediation process to broker the decision to hibernate. It is responsible for deciding how long a redundancy decision should persist and for informing layers of an impending hibernation before putting the hardware to sleep.

One possible approach to tackle the timing issue outlined previously, could be to set the periods individually for each layer. The node can be hibernated only for the duration of time that both layers are in sleep mode. This is problematic for a number of reasons, including the likelihood that one layer is active while another is sleeping, meaning that a higher layer may have to forward messages through a lower, sleeping layer. This would render the stack-based architecture for the protocols on a node, unsuitable. The mediator component, detailed next, could be retained in a variation to calculate the appropriate hibernation of the node based on the requested hibernation periods of the individual protocols. Indeed, we propose to keep such a mediation process for the MAS based architecture, detailed in Section 5.

4. PROVIDING A HIBERNATION POLICY THROUGH MEDIATION

One important aspect of the architecture in Figure 1, is the explicit separation of the decision to hibernate from the actual mechanics of performing the hibernation. This has a number of advantages. Consider the alternative where the layers would negotiate not only on whether to hibernate, but also on the duration of hibernation and potentially the components to shut down. When a new layer or protocol is introduced, for example where the application can veto the decision to hibernate, then it can get quite complex to deliver the additional co-operation required. This design simplifies the approach by the mediator storing each layers redundant/critical decision, Figure 2. When one changes, the mediator can examine all layers decisions and then act accordingly.

Within this design the layers do not interact, except to transmit messages between nodes through lower layers and this facilitates the replacement of one protocol with another without the need for changes to the surrounding layers. Therefore this approach gives flexibility in terms of adding additional hibernation protocols and replacing individual layers. Another point to note here is that the application must be able to deal with the hibernation and activation of the node. The base station must expect nodes to hibernate and thus receive no data from them on a temporary basis.

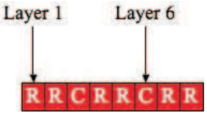

FIGURE 2. Mediation process for hibernating a node. A layer is either Redundant (R) or Critical (C).

Not only does the mediator decide on whether or not to hibernate, it also governs various temporal aspects such as how long a decision by a layer persists or whether it ever expires. It decides how long the node should hibernate for and it can also implement a number of different hibernation policies, including adaptive hibernation periods. Additionally, when layers operate at different frequencies in terms of redundancy evaluation, this layer can wake up individual layers without affecting dormant layers. This disjoint hibernation is useful when the frequency of evaluation affects a layers performance. Additionally, it could be used to optimally schedule layer activity and sleep time based on meta-information about a layers operation. Sleeping layers must, as described earlier, be capable of forwarding upper, active layers messages in order for it to function, which is undesirable and contrary to the notion of hibernation.

In addition, control of the time between the hibernate signal to layers and actual shutdown is located here, as this component is responsible for the actual shut down. Various shut down procedures are implemented, for instance some layers are given priority when shutting down. Certain layers will be given priority over other layers, thus giving them more time to inform their neighbours before the transceiver is powered off. In addition, the exact components to hibernate are controlled from the mediator [12]; substituting one policy for another is achieved without disruption to the other layers in many cases.

5. MULTI-AGENT SYSTEM APPROACH

The goal of our proposed solution is to allow two or more redundancy identification techniques to operate independently to hibernate a node without the need for interaction between protocols while one is asleep. Our proposed alternative is to transform the stack-based implementation into a set of cooperating agents on the WSN node, Figure 3. Essentially the IPM system is separated and behaves akin to an autonomic manager for the node. Interaction between the IPM and the rest of the system is limited to packet sends via the MAC and routing components, which could quite easily be implemented as agents. Further interaction only involves wakeup and sleep directives from the IPM. This means the users stack essentially operates in isolation and the MAS IPM could be considered as a middleware service, which maintains topology. Additionally, agents on a node can and must co-operate with their corresponding peer agent on a neighbouring node to reach their redundancy decision. This interaction is required to go through the user spaces routing and MAC layers, which never hibernate unless the node is asleep.

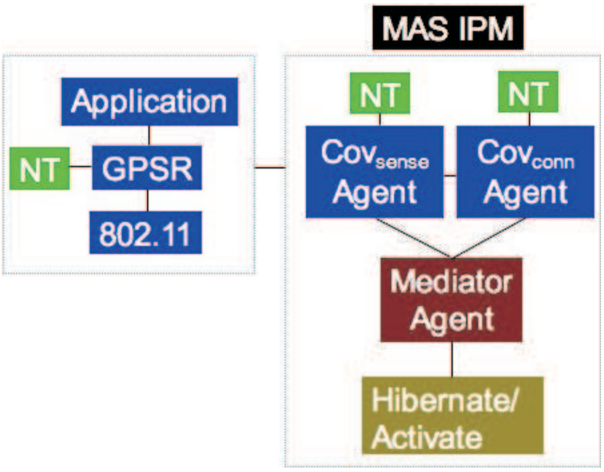

FIGURE 3. MAS based Intelligent Power Management system for WSNs. NT represents the agents neighbour table.

This architecture not only separates out the two distinct behaviours, but also facilitates the incorporation of additional power management agents without impacting the core WSN function. Under this approach timing issues can be mitigated and encapsulated in each agent, but it also ensures that the entities making the important hibernation decisions interact with each other rather than other aspects of the application. Removing the stack-based architecture removes the necessity for communication through a hibernating layer. When an agent is hibernating, it is now not visible to its other peers because it will not send or receive HELLO beacons. This in itself conserves both power and bandwidth. Finally, the mediation agent can act proactively in hibernating the hardware when all agents are hibernating. The duration is set to the next anticipated wake up time for a given agent. In this case all layers can be awoken or just the scheduled individual protocol.

6. RESOURCE BOUNDED REASONING

In deploying agents on sensor nodes, developers are faced with a number of problems, perhaps the most obvious is the limited spatiotemporal computational and power resources available. This paper advocates the use of agents, which are based on the Belief-Desire-Intention (BDI) [29] notion of agency. The BDI model acknowledges that agents are resource constrained and will be unable to achieve all of their desires even if their desires are consistent. An agent must fix upon a subset of desires within some intention selection process, and commit resources to achieving them. The implications of uncertainty, imperfect information, and limited resources are that it is not practical or even feasible for an agent or any other computational entity to find the optimal course of action. Such facets are abundantly present in WSNs from an unreliable communication channel to the severe computationally challenged nature of the devices. This raises a fundamental question: what does it mean for an agent to behave rationally when the agent does not have the resources or information to determine the course of action that yields the greatest utility?

In BDI logics, the concept of desire is a qualitative (binary) representation of utility. The intentions are chosen from among the desired states, using some metric that represents the utility value. In AFME [26], the metric for determining the utility is removed from the intention selection process and is placed within perceptors that generate beliefs about the costs and potential utility of certain actions. This is useful because it enables different metrics to be used for different commitments. The benefit and cost of certain actions will be dependent on context and variable beliefs provide a natural way of representing such possibly inaccurate data. Within the domain of WSNs, such inaccuracies are often present in the entries of a compiled neighbour table, for example. As stated previously, beacon messages can get lost and therefore neighbouring nodes will not be able to reason about their redundancy using accurate topology information.

In AFME, although agents adopt beliefs about the potential utilities of certain actions, it should be noted that the commitments chosen by the agent must still be from chosen among the desired states as determined by the rules governing the agent's behaviour. If a commitment is desired, then the potential utility value will represent its actual utility value within the intention selection process, otherwise the commitment is not considered for selection. The desires are still a qualitative or dichotomous representation of utility.

In reality, when someone is considering adopting a non trivial commitment, their beliefs about the costs and benefits of the commitment along with an abstract concept of the amount of resources available to them form an integral part of their reasoning process. Over time their beliefs about the costs and the benefits of adopting certain commitments change because they are dependent on context. When an agent is created in AFME, the total amount of resources available to the agent is specified so that the agent is aware of its limitations or constraints. Over time, these constraints will be altered and will consequently have an effect on the nature and degree of commitments adopted. For example, if the resources are related to the remaining battery power of the node, when the power drops, the agent will reduce the number of commitments it adopts. Alternately, the prevailing message latency in the locality of the agent could be used as a form of context. This context will cause the agent either to come online in order to balance the traffic more efficiently or to hibernate in order to remove its data from the channel.

The task of determining the subset of commitments to espouse or, in other words, the intention selection process is defined here as a classic 0-1 knapsack problem [30]. Given n items with corresponding profits p_j and weights w_j, the knapsack problem is the task of packing some of these items in a knapsack of capacity W, such that the profit sum of the included items is maximised. This is equivalent to the problem of an agent attempting to adopt the subset of commitments that maximise its utility with respect to its finite resources. The 0-1 knapsack problem is defined as follows:

maximise $\sum_{j=1}^{n} p_j x_j$ subject to $\sum_{j=1}^{n} w_j x_j <= W$, $x_j = 0$ or $1, j = 1...n$.

In attempting to determine an appropriate sleeping schedule for a protocol and an agent's deliberation process, we are faced with a difficult problem. Should an agent make a quick decision with a limited amount of data or allow the system to carry on operating as data is collected? An instance of this problem will be presented later in our experimental section, where selecting a balance between data volume and latency is critical in determining performance. This is somewhat similar to what Bellman referred to as the macroscopic principle of uncertainty in control theory [31]. It not only has implications on determining the optimal sleep time, but also on the collective behaviour of agents. For instance, in a given circumstance should an agent make a quick decision locally, possibly improving the responsiveness of the system, or act in a slower collaborative manner in making better informed decisions? This is an unanswerable question and is dependent on the application and the objectives that the agent is trying to achieve [32]. The use of the BDI model enables the construction of meta-level control mechanisms to allow agents to reason about the cost of decision making, e.g. transmission energy, and also in determining the responsiveness or sleep time.

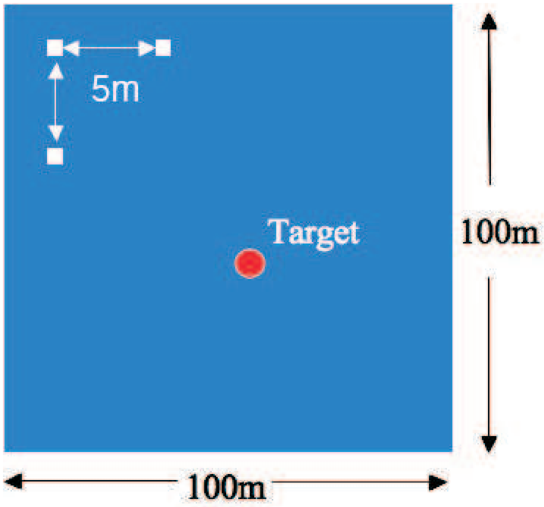

FIGURE 4. Experimental Setup: The deployment has a density of 1 node every 5m. The target is within the environment and the magnitude of the signal received at each sensor is inversely proportional to the distance to the target.

7. DISTRIBUTED CONSTRAINT OPTIMISATION

With the BDI model of agency and the localised knapsack problem, reasoning and resource management is primarily concerned with that of the individual. In this section, the focus is on collaborative behaviour concerning a team of agents. It should be noted, however, that in any collaborative activity, the behaviour and performance of the team ultimately comes down to the decisions made by the individual team members lest we contravene the notions of autonomy and rationality. Nonetheless, in situations where agents do decide to collaborate, it is essential that we have practical algorithms, as it were, to "lift the heavy stone". There are problems that exist that need to be solved in a distributed manner. For example, when a centralised approach is not practical or the agent has insufficient capabilities or knowledge to solve the problem in isolation.

In this section, we discuss distributed constraint optimisation. A Distributed Constraint OPtimization Problem (DCOP) is a constraint optimization problem that is solved in a distributed manner by a group of collaborating agents [11]. The agents share a common goal of choosing values for a set of variables such that the cost of a set of constraints over the variables is either minimized or maximized. A DCOP is defined as a tuple $\langle A, V, \mathcal{D}, f, \alpha, \sigma \rangle$, where:

A is a set of agents;

V is a set of variables, $\{v_1, v_2, \ldots, v_{|V|}\}$;

$\mathcal{D}$ is a set of domains, $\{D_1, D_2, \ldots, D_{|V|}\}$, where each $D \in \mathcal{D}$ is a finite set containing the values to which its associated variable are assigned;

f is a function $f : \bigcup_{S\in\mathcal{P}(V)} \prod_{v_i\in S} (\{v_i\} \times D_i) \to \mathbb{N} \cup \{\infty\}$ that maps variable assignments to costs;

α is a function $\alpha : V \to A$ that maps variables to agents. $\alpha(v_i) \mapsto a_j$ implies that agent aj assigns the value of variable vi;

σ is an operator that aggregates the individual f costs for the variable assignments. This is accomplished as follows: $\sigma(f) \mapsto \sum_{s\in\bigcup_{S\in\mathcal{P}(V)}\prod_{v_i\in S}(\{v_i\}\times D_i)} f(s)$.

The purpose of a DCOP algorithm is to enable each agent to assign values to their variables in order to either minimize or maximize $\sigma(f)$ for a given assignment. In Section 8, we discuss how a DCOP algorithm can be used to optimally manage the multiple sensor coverage and hibernation problem.

7.1. ADOPT Algorithm

In this section, we provide a high level and slightly simplified overview of the ADOPT algorithm, which is used to solve DCOPs. This will be sufficient for our purposes. An in depth discussion of the technical details of the algorithm goes beyond the scope of this article, but the interested reader is directed towards the preexisting literature [9] and the original ADOPT paper [33].

Initially in the ADOPT algorithm there is a preprocessing step. Within this step the constraint graph is converted into a constraint tree. The tree is constructed in such a manner that there are constraints only between a vertex and its ancestors or descendents.

As the algorithm is executing, every vertex of the constraint graph maintains (1) its current value, which is chosen from its domain, and (2) the values of its connected ancestors in the constraint tree, which are referred to as its current context. These values represent a partial solution of the DCOP. The vertices maintain for each value, the lower bounds on the cost of the solution that is consistent with the value and its current context. The lower bounds are initialized with the summation of the costs of the constraints between the connected ancestors. It is possible to determine these costs as the current context is known.

The vertices also maintain an upper bound on the cost of the solution that is consistent with their current context. The upper bound is initialised to infinity when the algorithm begins to operate. The lower bound of the current value of a vertex is referred to as its current lower bound. The smallest lower bound of all values is referred to as the best lower bound. The value associated with the best lower bound is referred to as the best value. When the algorithm begins to operate, the initial current value chosen by a vertex is the best value. The vertices also maintain a threshold value, which is initialised to zero. ADOPT maintains the following invariant in relation to the threshold: If it is lower than the best lower bound, it is increased to the best lower bound. If it is larger than the upper bound, it is reduced to the upper bound.

As the algorithm executes, if the current lower bound of a vertex is greater than the threshold, the current value is changed to the best value. Otherwise, the vertex keeps its current value. If the current value is changed, the vertex's connected descendents in the constraint tree are informed of its new value. The descendants perform similar computations, enabling the vertex to decrease its upper bound and increase its current lower bound. When the threshold of the root vertex of the tree is equal to its upper bound, the algorithm terminates.

As mentioned earlier if the current lower bound of the vertex is greater than the threshold, it changes its current value to its best value. There are two possible scenarios when this occurs.

1. If there are values whose lower bounds are less than the threshold, the best value is taken on and is kept until the lower bound of that value increases above the threshold. This process is repeated until all lower bounds are greater than or equal to the threshold. If this is the case, then the algorithm has reached the second possible scenario. It should be noted that during this first scenario, each value is only taken on once provided the ancestors do not switch values. The value is kept so long as the lower bound of the value is less than the threshold even if a different value has a smaller lower bound. This effectively represents a depth-first search.
2. If all lower bounds are greater than or equal to the threshold, the vertex increases the threshold to the best lower bound and then takes on its best value until the lower bound of that value increases. The procedure is then repeated. It should be noted that within this second scenario, the algorithm cannot return to the first scenario provided the vertex's ancestors do not switch values. In the second scenario, the algorithm always chooses the best value first; this procedure therefore represents a best-first search strategy.

7.2. CLDC Implementation

The current reference implementation of ADOPT [34] has been developed for simulation and within standard Java. In this section, an implementation of ADOPT that has been designed for the Constrained Limited Device Configuration (CLDC) subset of Java Micro Edition (JME) is discussed. CLDC is an extremely limited version of Java. It supports a very small subset of the standard Java classes along with some additional classes (such as those that form the Generic Connection Framework). There are also limitations on the Java Virtual Machine (JVM). For instance, the JVM has no floating point types, no object finalization or weak references, no Java Native Interface (JNI) support or reflection, no thread groups or daemon threads, and no application-defined class loaders. CLDC is the most widely used version of Java on mobile phones and WSN motes, such as the Sun SPOT and the Sentilla.

There are two reasons why the CLDC implementation of ADOPT was developed. (1) The current reference implementation has been designed for simulation. In this article, we consider the use of the algorithm for a real Sun SPOT WSN node deployment. (2) The current reference implementation has been designed for standard Java rather than CLDC. It therefore could not be used for the vast majority of mobile phones and Java-based WSN motes.

With the development of the CLDC implementation of ADOPT, a number of custom classes were created. The reason for this is that the reference implementation of ADOPT has dependencies on standard Java classes that are not available in CLDC. For instance, it uses the generic linked list class of Java. Creating customised classes provides a means to reduce the footprint and improve the maintainability of the software in that they need only meet the exact requirements of the problem to be addressed rather than provide a generic solution that can be used in a number of different cases. For example, methods return a specific class type rather than a generic object. This removes the need for casting.

In addition to re-implementing the algorithm such that it was compliant with CLDC, it was necessary to create classes that enable the nodes to communicate with each other over the radio channel. This functionality was implemented using the Sun SPOTs radio gram protocol, which facilitates datagram-based packet exchange.

The manner in which the CLDC implementation of ADOPT has been designed differs significantly from the original implementation. The design has been strongly influence by the 'Law of Demeter' (LoD) [35] or the principle of least knowledge. This specifies the coding guideline 'only talk to your immediate friends, not to strangers'. It requires that a method M of an object O only invokes the methods of the following objects: O itself, the parameters of M, and objects created or instantiated within M, and O's direct component objects. Developing code that conforms to the law tends improve the maintainability of the software and reduce the footprint by minimising code duplication [36, 37].

The CLDC implementation of ADOPT has been designed to be capable of operating in conjunction with AFME. As mentioned earlier, AFME is a reflective agent platform that has been designed for use with resource constrained devices. The idea is that AFME agents would use ADOPT to facilitate collaborative behaviour in situations where a particular problem has been formalised as a DCOP[1]. In such cases, AFME agents use ADOPT as a discrete collaborative action that is performed at a procedural level[2]. AFME agents use ADOPT by incorporating it as a service on the local platform. Although the ADOPT implementation has been designed to be compatible with AFME, it is quite possible to use ADOPT independently. This is useful in situations when there are not enough resources to operate both AFME and ADOPT. For instance, when using very low specification devices.

8. HETEROGENEOUS COVERAGE WITH NODE HIBERNATION

DCOP algorithms are designed to optimise multiple constraints in a distributed manner. Thus far in the article, we have only considered a coverage layer that comprises a network of homogeneous sensors. To illustrate the usefulness of DCOP algorithms, the manner by which they are used to optimally manage multiple heterogeneous coverage areas shall now be described.

In this article, we are considering WSNs that are densely populated and contain redundant nodes. In this section, we consider the situation in which each node has a number of sensing modalities. That is, each node has a number sensors on board that are capable of sensing different phenomena, such as light, temperature, and sound. Having a number of sensing modalities introduces a number of problems to our previously discussed algorithms for node hibernation, which were only considered from a homogeneous coverage perspective. When there are a number of different sensors on board a node, it has a number of different coverage areas for each different sensing modality. Additionally, it is conceivable that a node could have two different sensors for detecting the same phenomenon. For instance, it could have two light sensors one with a high power drain but large coverage area and one with a low power drain but lower coverage area. This case, however, is not considered here.

In this scenario, we initially model the situation as a graph colouring problem. Graph colouring is concerned with the assignment of labels (numbers), which are referred to as colours, to elements of a graph subject to certain constraints. The graph colouring problem is computationally hard. In its simplest form, it is a way of assigning numbers (which represent modalities in this example) to the nodes of a network such that no two adjacent nodes are assigned the same colour or modality in this case; this is commonly referred to as vertex colouring. Similarly, an edge colouring assigns a colour or number to each edge so that no two adjacent edges share the same value. In this article, we shall only consider vertex graph colouring.

In this application, there is an edge in the graph between two adjacent nodes if they lie within a specific threshold area. The threshold area in this case is less than the coverage area for the modality in question; there is a different threshold area for each different

[1] In this type of scenario, it is assumed that the DCOP is only part of the problem in that if it were the entire problem there would be no need for the AFME infrastructure.

[2] ADOPT is implemented at an imperative rather than declarative level.

coverage area. The idea is that in the WSN adjacent nodes in terms of the threshold area will not have the same sensing modality, but the overall coverage area will still be monitored in that the threshold area is less than the coverage area and the network is densely populated. Initially, the ADOPT algorithm is used to solve the graph colouring problem. A number of experiments have previously been conducted to examine how ADOPT solves the graph colouring problem and to measure its performance (see [33, 38] for more details). Once the ADOPT algorithm completes execution, each node in the network will have been assigned a colour; the modalities will be distributed throughout the network such that no two adjacent nodes have the same modality. Since we define the adjacency of nodes in terms of a threshold area that is less than the coverage area and the network is densely populated, the overall area should be reasonably well covered by each of the modalities.

When the initial phase of assigning nodes to modalities has completed, each set of nodes with the same modality is then treated as a separate network at a logical level in terms of determining node coverage redundancy. Even though adjacent nodes were not selected in the assignment process, there will still be redundancy and coverage overlap between nodes in that the area chosen for the threshold was less than the coverage area for each modality. The previous algorithm for detecting redundancy is then used to identify redundant nodes. A different mediator is required in this case, however. The reason for this is that the mediator must consider the overall picture in terms of connectivity and thus cannot be shut down based on the unilateral coverage area such as is the case in a homogeneous network. For instance, a node sensing one phenomenon could be used in terms of connectivity for transferring messages from nodes sensing a different phenomenon. This again illustrates the benefits of using the mediator approach.

9. EVALUATION AND DISCUSSION

In this section, we demonstrate the poor performance of the stack based architecture, primarily due to homogenous timing requirements. The strategy we adopt here is to specify optimal parameters for one of the layers, and measure the performance of the other protocol. We accomplish this by examining a number of QoS metrics associated with a WSN, namely message latency and application accuracy [39].

Additionally, we discuss the complexity of the CLDC ADOPT implementation and how it compares to the original implementation in terms of the software footprint.

9.1. Setup

The simulation environment used for our experimentation is J-Sim [28]. The simulated area for this set of experiments is defined as 100 meters x 100 meters with a deployed node density of one node every 5m, Figure 4. The result of this is that a fixed density of 400 nodes are used to cover the region of interest. One of the primary reasons for selecting this setup is to allow the results to be generalised to large areas by concatenation of networks similar to this. For example, a 500m x 500m region could be configured using 25 instances of the setup used here in a 5 x 5 grid formation. We place a target in the environment, which is given a power of 1000 units and decays according to the inverse square law of distance. The reason for having this stimulus within the environment is so that we can examine the performance of an application using the sensed data from the WSN. This decay model is applicable in many instances, including thermal radiation, light, sound and magnetic and gravitation fields, and has been used previously for similar experiments in [15]. It is initially located in the centre of the sensed area and takes a random walk around the area at a speed of 3 m/s. The results obtained here are the average of 5 individual executions of the simulation.

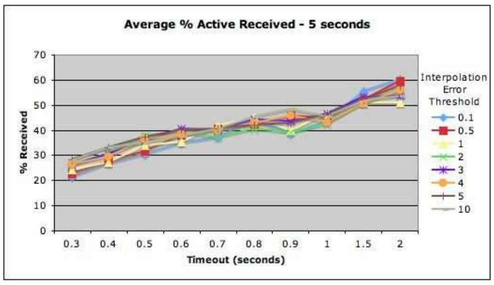

FIGURE 5. % of active nodes whose messages reach the base station within a certain timeout period as the interpolation error is varied. The active and sleep period is 5 seconds. Individual trends represent different interpolation thresholds set in IRISH.

The hibernation of nodes takes place using a combination of an interpolation based approach [17] for the sensing coverage component and CCP [3] with the sensing radius set to half the transmission radius. As proved in [5], such a strategy should guarantee connectivity indirectly by ensuring the region is sensing covered. The 802.11 MAC layer and GPSR that come with J-Sim complete the networking protocol stack for this experimentation. A short sleep and active time of 5 seconds is used, based on the results in [17] and finally, the application resident on the nodes sends its sensed data to the base station every 10 seconds. The next set of results show how this choice of parameters affects the message delivery and, to provide a better perspective on these results, the knock on impact on the performance of a target localisation application is also detailed.

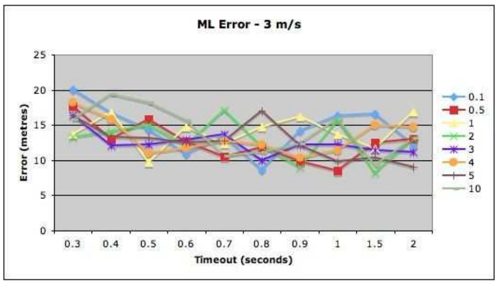

FIGURE 6. Localisation error for the ML technique as the density and latency are varied. The speed of the target is 3 m/s and the sleep/active period is 5 seconds. Individual trends represent different interpolation thresholds set in IRISH.

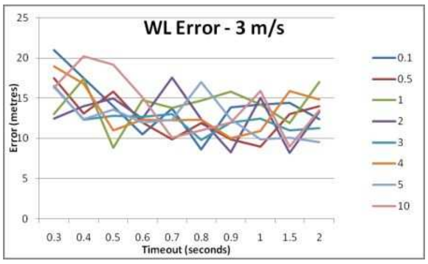

FIGURE 7. Localisation error for the WL technique as the density and latency are varied. The speed of the target is 3 m/s and the sleep/active period is 5 seconds. Individual trends represent different interpolation thresholds set in IRISH.

9.2. Results

The first of these results, Fig. 5, demonstrate the packet delivery over the active topology maintained by the connectivity preserving protocol and the IRISH protocol operating in tandem. The redundancy definition used by IRISH is based on the sensed data perceived by a node, if the neighbouring nodes can collectively interpolate its sensed value to within a predefined accuracy threshold then the node is deemed redundant. The higher the error tolerance the more nodes should be put to sleep, in theory. Figure 5 shows the significant delay in packets reaching the base station under the timing regime selected for optimal performance of the sensing redundancy identification technique. Even after 2 seconds, only 60% of messages transmitted had actually reached the base station in the best case.

In the next set of experiments we examine the implications of this poor routing performance on a possible WSN application, target localisation [40]. The task of target localisation, is to transform the streams of sensed data from the WSN into co-ordinates that pinpoint the location of a target in the sensed area. Two basic target localisation techniques are chosen for the application in this work, since they specifically do not require any prior characterisation of the target, making them applicable for many environments. They are the Weighted Average Localisation (WL) and the Maximum Signal Strength Localisation (ML) [17]. Target Localisation is a particularly suitable application for measuring the effect of latency, due to the continual motion of the target while messages are en-route. The greater the delay in messages reaching their destination, the greater the distance the target will have moved, which will contribute to the localisation error. This application is located at the base station and it waits an appropriate time to allow enough messages to be received at the base station before calculating the targets location. It cannot wait too long, as we have seen, so an optimal timeout value emerges that balances the requirement for a sufficient volume of fresh data.

Examining both Figures 6 and 7, we can see how negatively this timing regime impacts performance. Both techniques have a maximum precision of about 8m at about the 1 second timeout value. Given the size of the area to be monitored, this is quite a poor result. The reason behind this is the high frequency of evaluation of the coverage techniques. This yields a very dynamic topology over which messages take considerable time to travel. We may have adopted an alternative approach of selecting a longer sleep and active period. This would provide a stable topology of nodes, but it would not be able to respond to failed nodes in a timely fashion. More importantly, however, the IRISH protocols performance degrades as the sleep period increases, because the nodes cannot adapt to the changing sensed values due to the motion of the target. Using autonomous agents for the layers, it is hoped that different protocols can operate according to their own optimal schedule, which could dramatically increase the performance of the network.

9.3. ADOPT Performance

Distributed constraint optimisation is an NP hard problem. As discussed in Section 8, we use ADOPT to address the heterogeneous coverage problem. ADOPT is the first ever distributed, asynchronous, optimal algorithm for DCOP. The algorithm only requires polynomial space at each agent. One of the primary advantages of using ADOPT is that it contains an in built bounded error approximation mechanism. As the algorithm operates, the upper and lower bounds converge towards a solution. The algorithm need not continue operating until the threshold of the root vertex of the tree is equal to its upper bound, but when it is within a specific range. This provides a principled approach in the tradeoff between solution quality and resource usage.

In order to test the CLDC implementation of ADOPT, we replicated experiments developed for the reference implementation using the data set from [41]. The experiments were performed using both the reference implementation and the CLDC implementation [3]. As expected, the CLDC implementation provided the same results. From a practical perspective, however, the footprint of the CLDC version will be lower.

When considering the footprint of the software, we must also consider the footprint of other components it requires to execute. The current reference implementation of ADOPT has been developed for standard Java. The footprint of CLDC is considerably less that standard Java. This comes at a cost in terms of flexibility, however. For instance, the JVM of CLDC does not facilitate the dynamic loading of foreign objects. The reason for this is that within CLDC code must be preverified. This improves the performance of the JVM. The ADOPT algorithm does not require this functionality, thus this in no way inhibits its execution.

10. CONCLUSIONS

In this article, we considered how agent technology could be used to solve coverage and connectivity problems within a WSN, whereby different stake holders need to interact to achieve a global objective. When hibernating a node within a WSN, we must consider many factors, in particular connectivity and sensing coverage. A number of protocols exist for achieving each of these, however, little work has focussed on the issue of integrating these two necessary entities. In this paper, we detailed our initial stack-based solution to this problem and also the potential downfalls with this approach. In particular, differing evaluation periods, times for decisions and decision persistence mean that timing parameters must be strictly homogenous across all layers, leading to sub-optimal performance. Alternatively, if the layers are allowed to operate independently, a layer may forward its messages through a lower, hibernating layer.

Our solution to this was to recast the architecture to a MAS with each stake-holder realised as an agent. Now agents can operate proactively, and to some extent independently, according to their own schedule of hibernation and redundancy evaluation. For many combinations of protocols this could prove more efficient due to their conflicting timing desires. Our implementation to date has been focussed on the stack-based approach in the J-Sim WSN simulator [28]. We have completed an evaluation of a number of protocols in terms of their Energy-Density-Latency-Accuracy (EDLA) tradeoffs [42]. This range of QoS metrics are employed to ensure that a particular protocol or strategy does not artificially increase the longevity of the network by sacrificing the performance or accuracy of the application.

We discussed how the heterogeneous sensing coverage and node hibernation problem could be solved using a combination of the ADOPT algorithm, which is used to for distributed constraint optimisation, and the node hibernation algorithm. With this approach, ADOPT is first used to assign nodes to different sensing modalities. Subsequently, the node hibernation algorithm is used to identify redundant nodes and then power them down accordingly. Constraint optimisation is general is known to be NP Hard. ADOPT provides a principled bounded error approach that enables the tradeoffs between solution quality and resource consumption to be managed effectively. The current reference implementation of ADOPT is intended for simulation and has been designed for standard Java. The implementation discussed in this article has been developed for CLDC and has been tested on Sun SPOT motes.

11. FUNDING

This material is based on works supported by the Science Foundation Ireland under Grant No. 03/IN.3/I361. The authors would also like to acknowledge the support of the Irish Research Council for Science, Engineering and Technology.

REFERENCES

[1] Lin, K., Yu, J., Hsu, J., Zahedi, S., Lee, D., Friedman, J., Kansal, A., Raghunathan, V., and Srivastava, M. (2005) Heliomote: enabling long-lived sensor networks through solar energy harvesting. *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*, New York, NY, USA, pp. 309–309. ACM Press.

[2] Ferro Solutions Website. http://www.ferrosi.com.

[3] Xing, G., Wang, X., Zhang, Y., Lu, C., Pless, R., and Gill, C. (2005) Integrated coverage and connectivity configuration for energy conservation in sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, **1**, 36–72.

[4] Hill, J. (2003) System Architecture for Wireless Sensor Networks. PhD thesis UC Berkeley.

[5] Zhang, H. and Hou, J. C. (2005) Maintaining sensing coverage and connectivity in large sensor networks. *Ad Hoc & Sensor Wireless Networks (AHSWN)*, **1**, 89–124.

[6] Chen, B., Jamieson, K., Balakrishnan, H., and Morris, R. (2001) Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, Rome, Italy, pp. 85–96. ACM Press.

[7] Huang, C.-F. and Tseng, Y.-C. (2003) The coverage problem in a wireless sensor network. *WSNA '03: Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*, San Diego, CA, USA, pp. 115–121. ACM Press.

[8] Li, X.-Y., Wan, P.-J., and Frieder, O. (2003) Coverage in wireless ad-hoc sensor networks. *IEEE Transactions on Computers*, **52**, 753–763.

[9] Modi, P., Shen, W., Tambe, M., and Yokoo, M. (2006) Adopt: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence*, **161**, 149–180.

[3] These experiments were conducted on a desktop machine.

[10] Bowring, E., Tambe, M., and Yokoo, M. (2006) Multiply-constrained distributed constraint optimization. *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pp. 1413–1420. ACM New York, New York, USA.

[11] Shoham, Y., Leyton-Brown, K., and Leyton-Brown, K. (2008) *Multiagent systems: algorithmic, game-theoretic, and logical foundations.* Cambridge University Press.

[12] Sinha, A. and Chandrakasan, A. (2001) Dynamic power management in wireless sensor networks. *IEEE Design and Test of Computers*, **18**, 62–74.

[13] Marsh, D., Tynan, R., O'Kane, D., and O'Hare, G. (2004) Autonomic wireless sensor networks. *Engineering Applications of Artificial Intelligence*, **17**, 741–748.

[14] Huang, C.-F., Tseng, Y.-C., and Wu, H.-L. (2007) Distributed protocols for ensuring both coverage and connectivity of a wireless sensor network. *ACM Transaction on Sensor Networks (TOSN)*, **3**, 5.

[15] Karp, B. and Kung, H. T. (2000) Gpsr: greedy perimeter stateless routing for wireless networks. *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, New York, NY, USA, pp. 243–254. ACM Press.

[16] Xu, Y., Heidemann, J., and Estrin, D. (2001) Geography-informed energy conservation for ad hoc routing. *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, New York, NY, USA, July 16-21, pp. 70–84. ACM Press.

[17] Tynan, R., O'Hare, G. M. P., and Ruzzelli, A. (2006) Signal based node activation in wireless sensor networks. *AVSS '06: Proceedings of the IEEE International Conference on Video and Signal Based Surveillance*, Sydney, Australia 47. IEEE Computer Society.

[18] Qi, H., Wang, X., Iyengar, S. S., and Chakrabarty, K. (2001) Multisensor data fusion in distributed sensornetworks using mobile agents. *4th International Conference on Information Fusion*, Montreal, Canada.

[19] Braginsky, D. and Estrin, D. (2002) Rumor routing algorithm for sensor networks. *Wireless Sensor Networks and Applications (WSNA)*, Atlanta, Georgia, September, pp. 22–31.

[20] Sandhu, J., Agogino, A., and Agogino, A. (2004) Wireless sensor networks for commercial lighting control: Decision making with multi-agent systems. *AAAI Workshop on Sensor Networks*, San Jose, California.

[21] Fok, C.-L., Roman, G.-C., and Lu, C. (2005) Mobile agent middleware for sensor networks: An application case study. *In Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'05)*, Columbus, Ohio.

[22] Levis, P. and Culler, D. (2002) Mate : A virtual machine for tiny networked sensors. *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, San Jose, California.

[23] Adler, R., Flanigan, M., Huang, J., Kling, R., Kushalnagar, N., Nachman, L., Wan, C.-Y., and Yarvis, M. (2005) Intel mote 2: an advanced platform for demanding sensor network applications. *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*, New York, NY, USA, pp. 298–298. ACM Press.

[24] Simon, D. and Cifuentes, C. (2005) The squawk virtual machine: Java on the bare metal. *OOPSLA '05: Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, New York, NY, USA, pp. 150–151. ACM Press.

[25] Collier, R., O'Hare, G., Lowen, T., and Rooney, C. (2003) Beyond prototyping in the factory of the agents. *CEEMAS*, St. Petersburg, Russia.

[26] Muldoon, C., O'Hare, G. M. P., and Bradley, J. F. (2007) Towards reflective mobile agents for resource-constrained mobile devices. *AAMAS '07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, New York, NY, USA, pp. 1–3. ACM.

[27] Polastre, J., Hill, J., and Culler, D. (2004) Versatile low power media access for wireless sensor networks. *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, New York, NY, USA, pp. 95–107. ACM Press.

[28] Sobeih, A., Chen, W.-P., Hou, J. C., Kung, L.-C., Li, N., Lim, H., Tyan, H.-Y., and Zhang, H. (2005) J-sim: A simulation environment for wireless sensor networks. *anss*, Los Alamitos, CA, USA, pp. 175–187. IEEE Computer Society.

[29] Rao, A. S. and Georgeff, M. P. (1995) BDI Agents: from theory to practice. *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS'95)*, San Francisco, CA, June, pp. 312–319. AAAI.

[30] Dantzig, B. (1957) Discrete Variable Extremum Problems. *Operations Research*, **5**, 266–277.

[31] Bellman, R. E. (1968) *Some Vistas of Modern Mathematics.* University of Kentucky Press.

[32] Muldoon, C., O Hare, G., and O Grady, M. (2007) Managing Resources in Constrained Environments with Autonomous Agents. *Engineering Societies in the Agents' World*, Lecture Notes in Artificial Intelligence (LNAI), **4457**, pp. 320–339. Springer. This is an extended version of the ESAW'06 paper 'AFME: An Agent Platform for Resource Constrained Devices'.

[33] Modi, P., Shen, W., Tambe, M., and Yokoo, M. (2003) An asynchronous complete method for distributed constraint optimization. *Proceedings of Autonomous Agents and Multi-Agent Systems*, Victoria, Australia. ACM.

[34] Modi, P. (2009). USC DCOP repository.

[35] Lieberherr, K., Holland, I., and Riel, A. (1988) Object-oriented programming: An objective sense of style, . San Diego, CA, pp. 323–334. ACM.

[36] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., and Lorensen, W. (1991) *Object Oriented Modeling and Design.* Prentice Hall.

[37] Booch, G. (1994) *Object-oriented Analysis and Design, 2nd edition.* Addison Wesley.

[38] Yeoh, W., Koenig, S., and Felner, A. (2007) IDB-ADOPT: A Depth-First Search DCOP Algorithm. *IJCAI Distributed Constraint Reasoning Workshop*, Hyderabad, India.

[39] Adlakha, S., Ganeriwal, S., Schurgers, C., and Srivastava, M. B. (2003) Poster abstract: density, accuracy, delay and lifetime tradeoffs in wireless sensor networksa multidimensional design perspective. *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, New York, NY, USA, pp. 296–297. ACM Press.

[40] Brooks, R. R., Ramanathan, P., and Sayeed, A. M. (2003) Distributed target classification and tracking in sensor networks. *Proceedings of IEEE*, **91**, 1163 – 1171.

[41] Yin, Z. (2008). USC DCOP repository.

[42] Schurgers, C., Tsiatsis, V., Ganeriwal, S., and Srivastava, M. (2002) Optimizing sensor networks in the energy-latency-density design space. *IEEE Transactions on Mobile Computing*, **1**, 70–80.