A Distributed Coordination Framework for Wireless Sensor and Actor Networks

Tommaso Melodia, Dario Pompili, Vehbi C. Gungor and Ian F. Akyildiz

Broadband and Wireless Networking Laboratory
School of Electrical & Computer Engineering
Georgia Institute of Technology, Atlanta, GA 30332

{tommaso, dario, gungor, ian}@ece.gatech.edu

ABSTRACT

Wireless Sensor and Actor Networks (WSANs) are composed of a large number of heterogeneous nodes called *sensors* and *actors*. The collaborative operation of sensors enables the *distributed sensing* of a physical phenomenon, while the role of actors is to collect and process sensor data and perform appropriate actions.

In this paper, a coordination framework for WSANs is addressed. A new sensor-actor coordination model is proposed, based on an *event-driven clustering* paradigm in which cluster formation is triggered by an event so that clusters are created on-the-fly to optimally react to the event itself and provide the required reliability with minimum energy expenditure. The optimal solution is determined by mathematical programming and a distributed solution is also proposed. In addition, a new model for actor-actor coordination is introduced for a class of coordination problems in which the area to be acted upon is optimally split among different actors. An auction-based distributed solution of the problem is also presented.

Performance evaluation shows how global network objectives, such as compliance with real-time constraints and minimum energy consumption, can be reached in the proposed framework with simple interactions between sensors and actors that are suitable for large-scale networks of energy-constrained devices.

Categories and Subject Descriptors:
C.2.2 [Computer-Communication Networks]: Network Protocols-*routing protocols*

General Terms: Algorithms, Design, Reliability, Performance.

Keywords: Wireless Sensor and Actor Networks, Real-Time Communications, Energy Efficiency.

1. INTRODUCTION

Wireless Sensor and Actor Networks (WSANs) [1] are composed of a large number of heterogeneous nodes called *sensors* and *actors*. Sensors are low-cost, low-power, multi-functional devices that communicate untethered in short distances. The role of actors is to collect and process sensor data and perform appropriate actions. Hence, actors are resource-rich nodes equipped with better processing capabilities, higher transmission powers, and longer battery life.

In WSANs, a large number of sensor nodes, i.e., on the order of hundreds or thousands, are randomly deployed in a target area to perform a collaborative sensing task. Such a dense deployment is usually not necessary for actor nodes, since actors are sophisticated devices with higher capabilities that can act on large areas.

The collaborative operation of the sensor nodes enables the *distributed sensing* of a physical phenomenon. After sensors detect an event occurring in the environment, the event data is distributively processed and transmitted to the actors, which gather, process, and eventually reconstruct the event data. We refer to the process of establishing data paths between sensors and actors as *sensor-actor coordination*. Once an event has been detected, the actors coordinate with each other to make a decision on the most appropriate way to perform the action. We refer to this process as *actor-actor coordination*. As a result, the operation of a WSAN can be thought of to be an event-sensing, decision, and acting loop.

WSANs can be an integral part of systems such as battlefield surveillance, nuclear, biological or chemical attack detection, home automation and environmental monitoring [1]. For example, in fire detection applications, sensors can relay the exact origin and intensity of the fire to water sprinkler actors so that the fire can easily be extinguished before it spreads. Similarly, motion and light sensors in a building can detect the presence of intruders, and command cameras or other instrumentations to track them. Furthermore, sensors for structural health monitoring in airplanes or spaceships can drive instruments to timely take countermeasures against critical mechanical stress or structural faults. As a last example, in earthquake scenarios sensors can help locate survivors and guide robots performing rescue operations.

In a way, WSANs can be considered a distributed control system that needs to timely react to sensor information with an effective action. For this reason, *real-time coordination and communication* in WSANs is an important concern so as to guarantee timely execution of the right actions. Some recent papers [2][3][4] have considered the issue of real-time communication in sensor networks. However, as discussed in [5], there are still many open research challenges in order to enable real-time communication and coordination in sensor networks, especially due to resource constraints and scalability issues. Besides, none of these works deals with sensor-actor coordination. The *energy efficiency* of network communications is also crucial, since sensors are resource-constrained nodes with limited battery lifetime and communication capabilities [6]. Furthermore, sensor network protocols and algorithms should be *scalable* and *localized*, as the number of nodes can be arbitrarily high.

It has been recently shown in the literature that a fundamen-

MobiHoc'05, May 25–27, 2005, Urbana-Champaign, Illinois, USA.

tal trade-off exists between energy and latency for data delivery in sensor networks [7]. For this reason, in WSANs it is required to deliver event data in an energy-efficient way while respecting real-time delay constraints. As far as scalability is concerned, it has been pointed out [8][9] that the routing protocols that do not use geographical location information do not scale well in large networks. Conversely, the recent availability of small, inexpensive, and low-power GPS (Global Positioning System) receivers, together with techniques to deduce relative sensor coordinates from signal strengths [10], paved the way for *Geographical Routing* (also *Position Based Routing*) algorithms [11], which are becoming the most promising scalable solutions for critically energy-constrained sensor networks. Therefore, in this paper we study the *sensor-actor coordination* based on a geographical routing paradigm.

In order to guarantee scalability and energy efficiency, several solutions based on dividing the sensor network into different clusters have been proposed [12][13][14][15]. Most of the existing clustering algorithms can be classified as *topology dependent*, i.e., clusters are predetermined, depend on the topology of the sensor network, and may be adaptively reconfigured to deal with mobility or failure of the sensor nodes. Usually, in topology-dependent clustering one of the sensors is elected as a *cluster-head* in each cluster. When an event occurs, each sensor is already associated with a cluster-head.

In this paper, we propose to base the sensor-actor coordination on a new *event-driven clustering* paradigm, where cluster formation is triggered by an event so that clusters are created *on-the-fly* to optimally react to the event. In our approach, sensors detecting an event coordinate with each other so as to optimally associate each sensor with an actor. This way, only the event area is clustered, and each cluster consists of those sensor nodes that send their data to the same actor. Hence, the event information is collected at the optimal actor nodes while existing energy resources are better utilized, since clusters are formed only when necessary, based on the event features and on the position of the actors. The resulting architecture is shown in Fig. 1. The event-driven clustering approach also eliminates the communication overhead to maintain clusters before the event occurs, which is desirable especially in application scenarios where events are rare.

In addition, we introduce a model for actor-actor coordination. We define an optimization model for a class of coordination problems in which the area to be acted upon is optimally split among different actors depending on the actor capabilities.

The overall contribution of this paper is a comprehensive framework for coordination problems in WSANs, and can be outlined as:

Sensor-actor coordination:
- We define the event-driven clustering problem and determine the optimal solution by *Integer Linear Programming (ILP)* [16].
- We propose a multi-state distributed algorithm that achieves an energy-efficient solution for sensor-actor coordination and includes an adaptive mechanism that trades-off energy consumption for delay when event data must be delivered to the actors within predetermined latency bounds.

Actor-actor coordination:
- We define an optimization problem to divide the action workload among different actors, depending on the characteristics of the event, and formulate it as a *Mixed Integer Non-Linear Program (MINLP)* [17].
- We propose a distributed algorithm for the actor-actor coordination problem, based on localized auctions among the actors.

To the best of our knowledge, this is the first paper to comprehensively deal with integrated networks of sensors and actors, and to propose a unified framework for communication and coordination problems in WSANs. Since WSANs can enable a broad range of applications with different requirements, we focus on scenarios with immobile actors that can act on a limited area defined by their action range. The ultimate objective of this work is to demonstrate how global network objectives, such as compliance with real-time constraints and minimum energy consumption, can be reached in the proposed framework with simple interactions between sensors and actors, suitable for large-scale networks of energy-constrained devices.

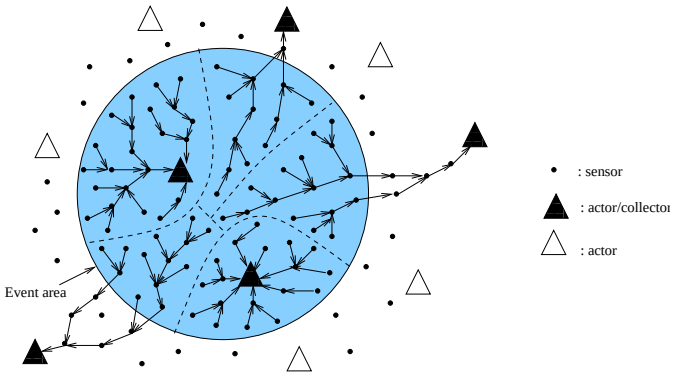

Figure 1: Event-driven clustering with multiple actors.

The remainder of this paper is organized as follows. In Sections 2 and 3 we discuss the sensor-actor coordination problem. In particular, in Section 2 we state the sensor-actor coordination problem and propose an integer linear programming formulation, while in Section 3 we describe a distributed protocol for online solution of the problem. In Sections 4 and 5 we discuss the actor-actor coordination problem. In particular, in Section 4 we state the actor-actor coordination problem, while in Section 5 we introduce a distributed solution based on a real-time localized auction mechanism. Detailed comparative performance evaluation and simulation results are presented in Section 6. Finally, in Section 7 we draw the main conclusions.

2. SENSOR-ACTOR COORDINATION: PROBLEM FORMULATION

Since, as discussed in the previous section, sensor-actor communications may have real-time requirements, we introduce a novel notion of reliability that accounts for the percentage of total received packets that are received within a pre-defined latency bound (which we refer to as *reliable packets*). Unlike other notions of reliability, the definition introduced here is related to the real-time delivery of data packets from sources to actors.

DEFINITION 1. *The* latency bound B *is the maximum allowed time between the instant when the physical features of the event are sampled by the sensors and the instant when the actor receives a data packet describing these event features. A data packet that does not meet the latency bound* B *when it is received by an actor is said to be* expired *and thus* unreliable. *Similarly, a data packet received within the latency bound is said to be* unexpired *and thus* reliable.

DEFINITION 2. *The* event reliability r *is the ratio of* reliable *data packets over all the packets received in a decision interval* [1]. *The* event reliability threshold r_{th} *is the minimum* event reliability

[1]Whenever a packet is dropped by an intermediate sensor, either because it violates the latency bound constraint or because of net-

required by the application. The lack of reliability *is the difference* $(r_{th} - r)$ *between the required event reliability threshold* r_{th} *and the observed event reliability* r *at a given time. A negative lack of reliability indicates a reliability above the required threshold and is also referred to as an* excess of reliability.

Note that the latency bound B and the event reliability threshold r_{th} are dependent on the application requirements.

The sensor-actor coordination problem consists of establishing data paths from each sensor residing in the event area to the actors by i) ensuring that the observed reliability r is above the event reliability threshold r_{th} (i.e., $r \geq r_{th}$); ii) minimizing the energy consumption associated with data delivery paths.

We refer to our solution for the sensor-actor coordination problem as *event-driven clustering with multiple actors* and model it as an Integer Linear Program (ILP). In Sections 2.1 and 2.2 we describe the network and energy model, respectively. In Section 2.3 we provide the complete ILP formulation of the problem.

2.1 Network Model

The network of sensor and actor nodes is represented as a graph $\mathcal{G}(\mathcal{S}^{\mathcal{V}}, \mathcal{S}^{\mathcal{E}})$, where $\mathcal{S}^{\mathcal{V}} = \{v_1, v_2, \ldots, v_N\}$ is a finite set of nodes (vertexes) in a finite-dimension terrain, with $N = |\mathcal{S}^{\mathcal{V}}|$, and $\mathcal{S}^{\mathcal{E}}$ is the set of links (edges) among nodes, i.e., $e_{ij} \in \mathcal{S}^{\mathcal{E}}$ iff nodes v_i and v_j (also i and j for simplicity in the following) are within each other's transmission range. Let $\mathcal{S}^{\mathcal{A}}$ represent the set of actors, with $N_A = |\mathcal{S}^{\mathcal{A}}|$. We refer to an actor that is collecting traffic from one or more sources as a *collector*. Let $\mathcal{S}^{\mathcal{S}}$ be the set of traffic sources, with $N_S = |\mathcal{S}^{\mathcal{S}}|$. This set represents the sensor nodes that detect the event, i.e., the sensors that reside in the event area. Since the set of sources is disjoint from the set of actors, $\mathcal{S}^{\mathcal{A}} \subset \mathcal{S}^{\mathcal{V}}$, $\mathcal{S}^{\mathcal{S}} \subset \mathcal{S}^{\mathcal{V}}$, and $\mathcal{S}^{\mathcal{A}} \cap \mathcal{S}^{\mathcal{S}} = \emptyset$.

2.2 Energy Model

An accurate model for energy consumption per bit at the physical layer is $E = E_{elec}^{trans} + \beta d^{\alpha} + E_{elec}^{rec}$, where E_{elec}^{trans} is a *distance independent* term that takes into account overheads of transmitter electronics (PLLs, VCOs, bias currents, etc.) and digital processing; E_{elec}^{rec} is a distance independent term that takes into account the overhead of receiver electronics, while βd^{α} accounts for the radiated power necessary to transmit one bit over a distance d between source and destination. As in [12], we assume that $E_{elec}^{trans} = E_{elec}^{rec} = E_{elec}$. Thus, the overall expression simplifies to $E = 2E_{elec} + \beta d^{\alpha}$, where α is the exponent of the path loss ($2 \leq \alpha \leq 5$), β is a constant $[Joule/(bits \cdot m^{\alpha})]$, and E_{elec} is the energy needed by the transceiver circuitry to transmit or receive one bit $[Joule/bits]$.

In our energy model we consider that, when a sensor node receives data from at least two other nodes, it aggregates the received information by *data fusion* [18], i.e., a single packet is created by merging multiple incoming packets, thus reducing the amount of data to be tranmsitted. To effectively support this function, an algorithm for data fusion should be implemented on each sensor, which is out of the scope of this paper. Moreover, we ignore the processing cost in our model, since the processing cost is much lower than the communication cost. This is justified by experimental results on sensor network prototypes such as [19], where the energy necessary to transmit 1 kbit is shown to be equivalent to the energy necessary to execute 300,000 processor instructions.

work or channel impairments, a short notification packet is sent to the actor, so that the lost packet can be taken into account in the computation of the reliability. Hence, in the definition "received packets" refers to the sum of data and notification packets

2.3 Integer Linear Program

The objective of the optimization problem is to find *data aggregation trees* (da-trees) from all the sensors that reside in the event area (referred to as sources) to the appropriate actors. A da-tree is composed by aggregating individual *flows*, where a flow is defined as a connection between a sensor and an actor. All leaves in a da-tree are sources (but not all sources are necessarily leaves), and each actor is either the root of a da-tree or does not participate in the communication. Da-trees are constructed in such a way that each source belongs to one tree only and each tree has only one actor as its root. Therefore, each source is associated with an actor to achieve an optimal strategy for event-driven clustering.

In fact, event-driven clustering can be seen as a joint twofold problem: i) select the optimal subset of actors to which sensor readings will be transmitted; ii) construct the minimum energy da-trees towards those selected actors that meet the required event reliability constraint. Each tree implicitly defines a cluster, which is constituted by all source nodes in the tree. Figure 1 gives an example of this configuration.

The optimal strategy for event-driven clustering is formulated as an *Integer Linear Program* (ILP) [16]. The network topology is assumed to be *1-connected*, i.e., at least one path exists between each sensor and actor. Note that this is not a strict requirement in dense sensor networks. We introduce the following notations that are used in the problem formulation:

- e_{ij} is a binary variable representing a link, that equals 1 iff nodes i and j are within each other's transmission range;
- c_{ij} is the cost of the link between nodes i and j, i.e., $2E_{elec} + \beta d_{ij}^{\alpha}$, where d_{ij} represents the distance between nodes i and j;
- x_{ij}^{k} is a binary variable that equals 1 iff link (i, j) is part of the da-tree associated with actor k;
- $f_{ij}^{k,s}$ is a binary variable that equals 1 iff node s sends data to actor k and link (i, j) is in the path from source s to actor k;
- $l^{k,s}$ is a binary variable that equals 1 iff sensor s sends data to actor k;
- p_{ij} is the propagation delay associated with link (i, j), defined as d_{ij}/v, where v is the signal propagation speed;
- $\tilde{d}$ is a parameter that accounts for the average sum of *processing*, *queuing*, and *access* delay at each sensor node;
- B is the latency bound on each source-actor flow;
- r and r_{th} are the event reliability and the required event reliability threshold, respectively;
- $b^{k,s}$ is a binary variable that equals 1 iff the connection between source s and actor k is not compliant with the latency bound, i.e., the end-to-end delay is higher than the latency bound B;
- Q is the number of non-compliant sources.

The problem can be cast as follows:

$\mathbf{P}_{Min}^{Com}$: Event-Driven Clustering with Multiple Actors

$$Given: \quad e_{ij},\ c_{ij},\ p_{ij},\ v,\ \tilde{d},\ B,\ r_{th} \tag{1}$$

$$Find: \quad x_{ij}^{k},\ f_{ij}^{k,s},\ l^{k,s},\ b^{k,s},\ r \tag{2}$$

$$Minimize: \quad C^{TOT} = \sum_{k \in \mathcal{S}^{\mathcal{A}}} \sum_{(i,j) \in \mathcal{S}^{\mathcal{E}}} x_{ij}^{k} \cdot c_{ij} + \gamma \cdot Q \tag{3}$$

$$Subject\ to:$$

$$\sum_{j \in \mathcal{S}^{\mathcal{V}}} (f_{sj}^{k,s} - f_{js}^{k,s}) = l^{k,s}, \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}; \tag{4}$$

$$\sum_{j \in \mathcal{S}^{\mathcal{V}}} (f_{kj}^{k,s} - f_{jk}^{k,s}) = -l^{k,s}, \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}; \tag{5}$$

$$\sum_{j \in \mathcal{S}^{\mathcal{V}}} (f_{ij}^{k,s} - f_{ji}^{k,s}) = 0,$$

$$\forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}, \forall i \in \mathcal{S}^{\mathcal{V}}\ s.t.\ i \neq s, i \neq k; \quad (6)$$

$$f_{ij}^{k,s} \leq e_{ij}, \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}, \forall i \in \mathcal{S}^{\mathcal{V}}, \forall j \in \mathcal{S}^{\mathcal{V}}; \quad (7)$$

$$f_{ij}^{k,s} \leq x_{ij}^{k}, \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}, \forall i \in \mathcal{S}^{\mathcal{V}}, \forall j \in \mathcal{S}^{\mathcal{V}}; \quad (8)$$

$$\sum_{k \in \mathcal{S}^{\mathcal{A}}} l^{k,s} = 1, \forall s \in \mathcal{S}^{\mathcal{S}}; \quad (9)$$

$$f_{ij}^{k,s} \leq l^{k,s}, \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}, \forall i \in \mathcal{S}^{\mathcal{V}}, \forall j \in \mathcal{S}^{\mathcal{V}}; \quad (10)$$

$$\varepsilon \cdot [B - \sum_{(i,j) \in \mathcal{S}^{\mathcal{E}}} f_{ij}^{k,s}(p_{ij} + \tilde{d})] \leq b^{k,s},\ \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}; \quad (11)$$

$$Q = \sum_{k \in \mathcal{S}^{\mathcal{A}}} \sum_{s \in \mathcal{S}^{\mathcal{S}}} b^{k,s}; \qquad r = \frac{|\mathcal{S}^{\mathcal{S}}| - Q}{|\mathcal{S}^{\mathcal{S}}|} \geq r_{th}. \quad (12)$$

The objective function in (3) minimizes the overall energy consumption and imposes a penalty by multiplying the number Q of non-compliant sources by a penalty coefficient γ whose value must be high enough (e.g., orders of magnitude higher than the energy consumption) to guarantee uniqueness of the solution. This allows minimizing the number of non-compliant sources Q (eq. 12) with a single-objective problem. As previously discussed, a flow is a connection between a source and an actor. Flows associated with the same actor are aggregated in a da-tree. Constraints (4), (5), and (6) express conservation of flows [16], i.e., each source generates a flow, which is collected by an actor. In particular, constraint (4) guarantees that a source node generates a flow on the tree of the selected actor, and only on that one; while non-source nodes do not generate any flow. Constraint (5) requires that flows generated by each source be collected by one actor only. Constraint (6) imposes that the balance between incoming and outgoing flows is null for non-source and non-actor nodes. Constraint (7) ensures that flows are created on links between adjacent nodes (i.e., that are within transmission range of each other). Constraint (8) forces all flows from different sources but directed towards the same actor to be aggregated in the tree associated with that actor. Constraint (9) imposes that each source send data to exactly one actor. Constraint (10) ensures that all flow variables from a source to a particular actor are zero unless that actor is selected by the source. Constraint (11) requires that the binary variable $b^{k,s}$ be equal to 1 always and only when the flow between source s and actor k violates the latency bound B. Note that the small negative coefficient ε is needed to scale the value of the difference between the delay and the delay bound to make it smaller than 1. In (12), Q is defined as the number of non-compliant sources and the reliability r is constrained to be over the required threshold.

It can be shown that the problem $\mathbf{P}_{Min}^{Com}$ is at least as complex as the Geometric Connected Dominating Set problem, which is proven to be NP-complete [20]. Hence, $\mathbf{P}_{Min}^{Com}$ is NP-complete. However, it is still possible to solve $\mathbf{P}_{Min}^{Com}$ for networks of moderate size (up to 100 nodes), as will be shown in Section 6. This allows gaining some insight on the properties of the optimal solution, and using it as a benchmark for the performance of suboptimal, but more scalable algorithms, such as that introduced in the following section.

3. SENSOR-ACTOR COORDINATION: DISTRIBUTED PROTOCOL

In this section, we introduce a scalable and distributed protocol to addresses the sensor-actor coordination problem in WSANs. The objective of the protocol is to build energy-efficient da-trees between the sources that reside in the event area and the actors, in order to provide the required reliability r_{th} with minimum energy expenditure. The proposed protocol constructs da-trees between sources and actors that can be seen as an approximate solution for the event-driven clustering with multiple actors problem, described in Section 2.

As discussed in [21], in geographical routing algorithms *localized routing decisions*, i.e., based on localized topology information, can lead to data paths whose energy efficiency is close to the global optimum. This means that in densely deployed sensor networks topology information related to network regions that are "far" from where the routing decision is being taken are not essential. In fact, this information does not considerably influence the energy efficiency of the overall data path. For this reason, the objective of the proposed protocol is to minimize the energy consumption by relying on localized information and on *greedy* routing decisions. Conversely, complying with pre-determined delay bounds requires some form of end-to-end feedback. Hence, the proposed protocol favors local behavior for each individual sensor node that results in a global network behavior that is compliant with the application requirements, i.e., provide an event reliability r above the required threshold r_{th} (defined in Section 2) and minimize the energy consumption. This is achieved by relying on feedback information from the actors/collectors.

In the description of the protocol, we assume that each sensor node is aware of: i) its position, as the sensor node can be equipped with a GPS receiver or the position can be determined by means of localization techniques [10]; ii) the position of its neighbors, as every node periodically sends its position to its neighbors; iii) the position of the actors, as each actor periodically beacons its position in the sensor field; iv) the network is synchronized by means of one of the existing synchronization protocols [22]. A study on the impact of localization and synchronization errors is left for future work.

An important issue in geographical routing algorithms is to avoid the creation of loops in the data paths. Hence, before proceeding with an overview of the proposed protocol, we introduce some concepts related to *path loop freedom* that will be used in the description of the sensor-actor coordination framework.

DEFINITION 3. *Given nodes v and x, the* absolute advance *of node x, with respect to v, is the distance between v and its closest actor c_v minus the distance between x and its closest actor c_x*[2].

DEFINITION 4. *Given nodes v and x, the* advance towards the collector c *of x, with respect to v, is the distance between v and c minus the distance between x and c.*

Intuitively, if x has *positive absolute advance* with respect to v, it means that x is closer than v to one (whatever) actor. If x has *positive advance towards collector* c with respect to v, x is closer than v to actor c. For any multi-hop path, a *positive absolute advance* at every hop guarantees loop freedom, irrespective of the final destination, since at each hop the packet is closer to a collector than at the previous hop. A *positive advance towards an actor* c at every hop guarantees a loop-free path from a source node to the actor c.

Most of the prior research in geographical routing protocols assumes that nodes can either work in a *greedy mode* or in a *recovery mode*. When in greedy mode, the node that currently holds the message tries to forward it towards the destination. The recovery mode

[2]Note that c_v and c_x can be different actors.

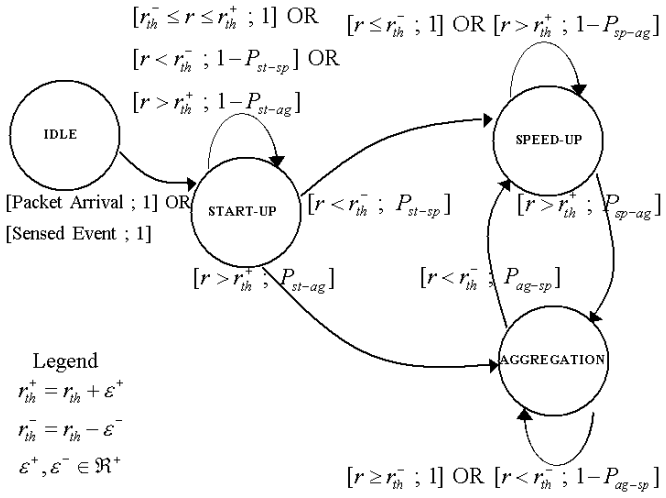

Figure 2: State transition diagram for a sensor node.

is entered when a node fails to forward a message in the greedy mode, since none of its neighbors is a feasible next hop. Usually this occurs when the node observes a void region between itself and the destination. Such a node is referred to as *concave* node. Recovery mechanisms, which allow a packet to be forwarded to the destination when a concave node is reached, are out of the scope of this paper, since sensor networks are usually high density networks, and the probability of encountering void regions is very low. For this reason, the protocol proposed in this section assumes that no void regions exist, although it can be enhanced by combining it with one of the existing recovery mechanisms (e.g., [23]).

The objective of the proposed protocol is to create da-trees between the sources and a subset of the actors, referred to as collectors. A da-tree is thus created between each collector and the sources associated with that collector. This way, the set of sources is implicitly clustered, each cluster being composed of the sources associated with a single collector.

3.1 Overview of the Multi-State Protocol

Each sensor alternates among four different states, namely *idle*, *start-up*, *speed-up*, and *aggregation state*. An overview of the state transitions is depicted in Fig. 2. The main objective of these state transitions is to reduce the number of hops when the reliability requirement is violated and to save energy when the reliability requirement is met.

Source nodes add a timestamp value to the event data packet that they transmit to the actors, to allow the corresponding actors to compute the delay of each packet. For each decision interval, each actor then computes the event reliability r as the ratio of unexpired packets over all received packets (i.e., data and notifications) and periodically broadcasts its value. Sensor nodes associated with that collector base their state transitions on the reliability observed by the collector, which is broadcast at the end of each decision interval. When the advertised value r is below the so-called *low event reliability threshold* r_{th}^-, where $r_{th}^- = r_{th} - \epsilon^-$, i.e., the lack of reliability ($r_{th} - r$) is above a certain positive margin ϵ^-, then it is necessary to speed-up the data delivery process by reducing the end-to-end delay. Conversely, when the advertised value r is above the so-called *high event reliability threshold* r_{th}^+, where $r_{th}^+ = r_{th} + \epsilon^+$, i.e., the excess of reliability ($r - r_{th}$) is above a certain margin ϵ^+, then there is reliability in excess that can be traded off for energy savings. Note that the small coefficients ϵ^+ and ϵ^- are needed to define a tolerance zone around the required reliability threshold for practical purposes (i.e., avoid instability). Good values for ϵ^+ and ϵ^- find a good compromise between stability and tolerance and can be determined by simulation.

Each sensor node starts in an $idle$ state, where it samples the environment for occurring events and monitors the channel for incoming data packets. A sensor enters the start-up state when it either senses an event or it receives the first data packet from a neighboring sensor. The collective operation of sensor nodes in the start-up state allows to timely establish paths to an actor for each source that resides in the event area. The paths established constitute a good compromise between latency and energy consumption.

Sensor nodes listen for feedback from the collector/actor they are associated with. If the event reliability r is below the low event reliability threshold r_{th}^-, it is necessary to reduce the sensor-to-actor delay, by reducing the end-to-end path length. Hence, when the reliability is advertised to be below the low reliability threshold ($r < r_{th}^-$), a sensor in the start-up state enters the speed-up state with probability P_{st-sp}, which is a monotonically increasing function of the *lack of reliability*. The notation $[cond; P]$ in Fig. 2 indicates a transition occurring with probability P when the condition $cond$ is verified.

If the event reliability r is above the high event reliability threshold r_{th}^+ (i.e., $r > r_{th}^+$), it is possible to save energy. Hence, a node in the start-up state enters the aggregation state with probability P_{st-ag}, which is a monotonically increasing function of the *excess of reliability*, where it tries to minimize the energy consumption associated with its transmission by relaying data to the closest neighbor that participates in a da-tree.

Then, sensors can alternate between the speed-up and the aggregation state in order to respond to feedbacks from collectors. Hence, as shown in Fig. 2, a sensor in the speed-up state enters the aggregation state with probability P_{sp-ag} when $r > r_{th}^+$, while a sensor in the aggregation state enters the speed-up state with probability P_{ag-sp} when $r < r_{th}^-$. P_{ag-sp} increases with increasing advertised lack of reliability, while P_{sp-ag} increases with increasing excess of reliability. The objective of our proposed protocol is to converge to a solution with reliability close to the event reliability threshold and minimal energy consumption.

Note that probabilistic policy prevents system oscillations that would occur if all sensors changed state at the same time. Clearly, the proposed algorithm is particularly effective when the time needed to set-up the tree is small as compared to the length of the subsequent monitoring and acting phase.

In the following we describe the operations of each state.

3.2 Start-up State

As shown in Fig. 2, a node enters the start-up state from the idle state when it detects an event, or when it receives a packet to be relayed to an actor.

The sensor node i in the start-up state, either as a source or as a relayer for a data packet, selects its next-hop based of the so-called *two-hop rule*. According to the two-hop rule, node i selects as next hop among its neighbors the node j that minimizes the sum of the energy consumption from i to j and the energy consumption from j to the actor closest to j, which is computed according to the energy model introduced in Section 2.2. Hence, the energy consumption E_j associated with a neighbor j of i is

$$E_j = 2E_{elec} + \beta d_{ij}^{\alpha} + 2E_{elec} + \beta d_{jc_j}^{\alpha}, \qquad (13)$$

where d_{ij} represents the distance between i and j while d_{jc_j} repre-

Algorithm 1 Start-up State

Pseudo-code executed by a generic node v in the start-up state
$mincost = \infty$
if (($I\_am\_a\_source$) OR ($I\_am\_a\_relayer$)) **then**
  for each of my neighbors N_i **do**
    for each actor s_k **do**
      if cost(v,N_i)+cost(N_i,s_k) $< mincost$ **then**
        $mincost$=cost(v,N_i)+cost(N_i,s_k)
        $nexthop$=N_i
      end if
    end for
  end for
end if
Inform $nexthop$ that it is a relayer

sents the distance between j and its closest actor c_j. The two-hop rule selects as next hop the node j associated with the minimum two-hop energy consumption. As a result, the source-actor path will be established by applying the two-hop rule iteratively. Noticeably, since at each step the routing decision is independent of previous decisions, a packet intended for a certain actor by its generating source can be transmitted towards another destination actor by an intermediate node in the end-to-end path. For this reason, the collector actor transmits its identifier on the reverse da-tree in order to inform the source sensors about its own identity. The operations executed by a sensor node in the start-up state are detailed in Algorithm 1.

The two-hop rule produces loop-free paths, as stated below.

LEMMA 1. *A next hop selected with the* two-hop rule *has a* positive absolute advance *(see Def. 3).*

As a consequence, the two-hop rule produces a loop-free path between source and actor. We omit the proof of Lemma 1 because of page constraints.

3.3 Speed-up State

The objective of the collaborative operation of nodes in the speed-up state is to minimize the number of hops between sources and actors. This is achieved by applying the Greedy Routing Scheme (GRS) [24] forwarding rule. According to GRS, each node sends the packet to the node closest to the destination within the transmission range. It is intuitive that this rule minimizes the number of hops in the path, the distance travelled by the packet, and the number of transmissions of the same data packet, along with the channel utilization. The pseudo-code of the operations executed by a sensor node in the speed-up state is reported in Algorithm 2.

Algorithm 2 Speed-up State

Pseudo-code executed by a generic node v in the speed-up state
for each neighbor N_i **do**
  if (distance(v,N_i)> distance(v,$next\_hop$)) **then**
    $next\_hop = N_i$
  end if
end for

3.4 Aggregation State

The objective of the aggregation state is to reduce the overall energy consumption. To this end, sensor nodes in the aggregation state take routing decisions that reduce the global energy consumption, by relying on the data fusion algorithm that we assume to be implemented on each sensor. Since data packets can be aggregated by any node in the network, the objective of a node in the aggregation state is to route data to the closest node in its neighborhood that is part of da-tree. This way, the incremental energy consumption to collect the information from the considered sensor is minimized.

As previously discussed, after da-trees are established, each sensor knows which collector-actor it is associated with. By overhearing transmissions on the shared medium, each sensor learns which of its neighbors are part of a da-tree (as some neighbor sensors may not even be in the event-area) and which da-tree (if any) are they part of, i.e., which collector actor are they associated with. Hence, node v in the aggregation state first evaluates the cost of transmitting data to those among its neighbors that are part of a da-tree. This way, it can identify a minimum-cost neighbor, i.e., the neighbor v_{min} that requires minimum energy consumption to be reached among those associated with one of the da-trees. Two different situations can occur. The node v_{min} can either be on the same da-tree as v, and hence associated with the same collector; or it can be in a different da-tree.

If v_{min} is in the same da-tree as v, v_{min} can be selected as next hop by v only if it has a positive advance towards the collector that both nodes are associated with, i.e., if v_{min} is closer than v to the collector (see Def. 4). This guarantees loop freedom. In the resulting da-tree, every parent node is assured to have positive advance towards the collector, with respect to each child. When v_{min} is selected, the individual transmission cost for v is locally minimized and the overall cost of the tree is thus reduced. If v_{min} has no positive advance towards the collector with respect to v, v deletes v_{min} from the list of possible next hops and determines a new v_{min} among the remaining neighbors.

The other possible situation occurs when v_{min} is associated with a different collector than v, i.e., v and v_{min} are in two different da-trees. In this case, v is allowed to select v_{min} as its next hop only if v is a leaf in its da-tree and v_{min} has a positive advance towards its actor with respect to v. This guarantees loop-freedom of the overall tree, as every parent node is assured to have a positive advance towards the actor with respect to each child. Conversely, it can be easily shown that if non-leaf nodes are allowed to switch from one da-tree to another, loops may be created, as the condition that every parent node is closer to the actor than each child does not necessarily hold anymore. The detailed operations executed by a sensor node in the aggregation state are reported in Algorithm 3.

Algorithm 3 Aggregation State

Pseudo-code executed by a node v in the aggregation state
for each of my transmitting neighbors N_i **do**
  if (cost(v,N_i) < cost(v,$nexthop$)) **then**
    v_{min}=N_i
  end if
end for
s=actor(v_{min})
if ($s == myactor$) **then**
  if distance(v_{min},s)<distance(v,s) **then**
    $nexthop$=v_{min}
  else
    delete v_{min} from list and restart Aggregation State
  end if
else if $I\_am\_a\_leaf$ **then**
  $nexthop$=v_{min}
else
  delete v_{min} from list and restart Aggregation State
end if

4. ACTOR-ACTOR COORDINATION: PROBLEM FORMULATION

The objective of actor-actor coordination is to select the best actor(s) to perform appropriate action on the event area. At the end of the sensor-actor coordination phase described in Section 3, one or multiple actors, which we denote as *collectors*, receive sensor readings from sources that sense the event. These sources define the *event area*. The event area corresponds to the *action area*, i.e., the area where the actors should act. In particular, each collector receives data from a subset of the sources (*cluster* in Section 2). Each cluster area identifies a portion of the action/event area and is under the responsibility of the corresponding collector. However, the collector may not be able to act on its entire responsible area, i.e., this area may not be totally within the collector's *action range*. The action range defines the circular area where an actor is able to act. Moreover, the collector may not be the "best" actor for that task in terms of *action completion time* and/or *energy consumption*, where the former is the time to perform the action and the latter is the required energy for the action. For these reasons, actor-actor coordination is required before initiating actions.

DEFINITION 5. *The* action completion bound *is the maximum allowed time from the instant when the event is sensed to the instant when the action is completed.*

The coordination objective of each collector actor is to find the optimal actors to timely act on the portion of the event area under its own responsibility. In particular, if multiple actors can act on a certain area we refer to the area as an *overlapping area* (region areas numbered from 1 to 8 in Fig. 3). On an overlapping area the actor-actor coordination problem consists of selecting a subset of the actors and their action powers to optimally divide the action workload, so as to maximize the *residual energy*[3] of the actors while respecting the *action completion bound*, in order to extend the lifetime of the actors. We refer to an area where only one actor can act as *non-overlapping area* (unshaded regions in Fig. 3). For such an area, the coordination problem simplifies to selecting the power level for the actor that minimizes the energy consumption while respecting the action completion bound. For this reason, we assume that the coordination problem involves only overlapping areas and that the available energy of each actor is already discounted with the energy needed to act on non-overlapping areas.

In Section 4.1 we present the action area model and the actor model. These models are then used in the formulation of the problem in Section 4.2.

4.1 Action Area and Actor Model

Let $\mathcal{S}^A$ be the set of actors, with $N_A = |\mathcal{S}^A|$, and let $\mathcal{S}^C$ be the set of collectors ($\mathcal{S}^C \subseteq \mathcal{S}^A$). As mentioned before, collectors receive data from sources (sensors), and from the source positions they can identify the portion of the whole event area they are responsible for. By referring to Fig. 3, we introduce the following notations:

- $\mathcal{A}^D$ is the whole deployment area, which can be modelled as a *grid* of square cells.
- Actor $a \in \mathcal{S}^A$ has coordinates $C_a^A = (x_a^A, y_a^A)$ and action range R_a, which defines the circular area $\mathcal{A}_a^A$ where it can act;
- $\mathcal{A}_c^C$ is the cluster area that is under the responsibility of *collector* $c \in \mathcal{S}^C$, and the whole event area is thus $\bigcup_{c=1}^{|\mathcal{S}^C|} \mathcal{A}_c^C$;

[3]Although actors are resource-rich nodes, the order of magnitude of the energy required for actions is higher than that required for communication. Hence, it is important to save action energy in order to extend the lifetime of actors.

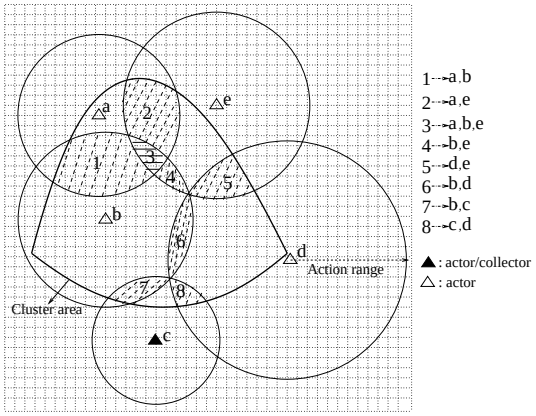

Figure 3: Cluster area for collector c.

- $\mathcal{A}_{c,nov}^{C,h}$ and $\mathcal{A}_{c,ov}^{C,m}$ are the h^{th} *non-overlapping* and the m^{th} *overlapping* areas, respectively, inside the portion of area under the responsibility of collector c. H_c represents the number of non-overlapping areas, while M_c represents the number of overlapping areas;
- $\mathcal{S}_{c,ov}^{A,m}$ is the set of actors that can act on the m^{th} overlapping area $\mathcal{A}_{c,ov}^{C,m}$ that is under the responsibility of collector c.

Each actor a is characterized by the following parameters:
- R_a $[m]$ is the action range of a;
- P_a^{Max} $[W]$ is the maximum power that actor a can use to perform the action. Actors can select their power among L different levels

$$P_{a,p} = \frac{P_a^{Max}}{L} \cdot p,\ p = 1, 2, \ldots, L \tag{14}$$

where $P_{a,p}$ is the p^{th} power level for actor a. As will be shown in (15), a higher power corresponds to a lower action completion time;
- η_a is the *efficiency* of actor a (see (15));
- $E_a^{Av}[J]$ is the available energy of actor a.

4.2 Mixed Integer Non-Linear Program

In this section we formulate the actor-actor coordination problem as a *Mixed Integer Non-Linear Program* (MINLP). The objective is to find, for each portion of the event area, the subset of actors that maximizes the average residual energy of all actors involved in the action, under the constraint of meeting the action completion bound. We define the problem according to the following assumptions: i) the energy required to perform the action is orders of magnitude higher than the energy required for communication; ii) actors are able to *selectively* act on part of their action area, i.e., if actor a is chosen to act either on an overlapping or non-overlapping area, this does not imply that it must act on the entire area $\mathcal{A}_a^A$ in its action range R_a.

Let us introduce the following notations:
- $P_{a,p}^{(m)}[W]$ is the p^{th} power level of actor a for the m^{th} overlapping area $\mathcal{A}_{c,ov}^{C,m}$, whose measure is $A_{c,ov}^{C,m}$ $[m^2]$;
- $\underline{X}^{(m)}$ is a binary matrix whose element $[x_{a,p}^{(m)}]$ is equal to 1 iff actor a acts on the overlapping area $\mathcal{A}_{c,ov}^{C,m}$ using power level $P_{a,p}^{(m)}$;
- $T_{a,p}^{(m)}$ $[s]$ is the action completion time for actor a, characterized by efficiency η_a, on the m^{th} overlapping area, when the actor uses

the p^{th} power level

$$T_{a,p}^{(m)} = K \cdot \frac{A_{c,ov}^{C,m}}{\eta_a \cdot (P_{a,p}^{(m)})^{\gamma}}, \quad (15)$$

where K $[W^{\gamma} \cdot s/m^2]$ is a constant, and γ is an a-dimensional parameter ranging in $(0, 1]$, which captures power inefficiencies;
- δ_c $[s]$ is the *action completion bound* (i.e., the maximum time for the action to be completed). This depends on the event, on the collector c, and on the application;
- $I_a^{(m)}$ is equal to 1 iff the m^{th} overlapping area is in the action range of actor a, 0 otherwise;
- h_a is a binary variable equal to 1 iff actor a is involved in an action.

We can now formulate the optimization problem as follows:
$\mathbf{P}_{Max}^{Res}$: **Residual Energy Maximization Problem**

$$Given: \quad N_A,\ L, M_c, E_a^{Av},\ T_{a,p}^{(m)}, I_a^{(m)} \quad (16)$$

$$Find: \quad \underline{X}^{(m)} = [x_{a,p}^{(m)}],\ h_a \quad (17)$$

$$Maximize: \quad E_{Avg}^{Res} = \frac{\sum_{a=1}^{N_A} h_a E_a^{Res}}{\sum_{a=1}^{N_A} h_a} \quad (18)$$

$$Subject\ to:$$

$$E_a^{Res} = E_a^{Av} - E_a^{Req} \geq 0,\ \forall a; \quad (19)$$

$$E_a^{Req} = \sum_{m=1}^{M_c} \left(\frac{\sum_{p=1}^{L} x_{a,p}^{(m)} P_{a,p}^{(m)}}{\sum_{a=1}^{N_A} \sum_{p=1}^{L} \frac{x_{a,p}^{(m)}}{T_{a,p}^{(m)}}} \right),\ \forall a; \quad (20)$$

$$\sum_{p=1}^{L} x_{a,p}^{(m)} \leq 1,\ \forall a,\ \forall m; \quad \sum_{a=1}^{N_A} \sum_{p=1}^{L} x_{a,p}^{(m)} \geq 1,\ \forall m; \quad (21)$$

$$\frac{1}{\sum_{a=1}^{N_A} \sum_{p=1}^{L} \frac{x_{a,p}^{(m)}}{T_{a,p}^{(m)}}} \leq \delta_c,\ \forall m; \quad (22)$$

$$h_a \leq \sum_{p=1}^{L} \sum_{m=1}^{M_c} x_{a,p}^{(m)},\ \forall a; \quad h_a \geq x_{a,p}^{(m)},\ \forall a,\ \forall p,\ \forall m; \quad (23)$$

$$x_{a,p}^{(m)} \leq I_a^{(m)},\ \forall a,\ \forall p,\ \forall m. \quad (24)$$

Constraint (19) guarantees a non-negative residual energy for each actor. Constraint (20) defines the energy required for actor a to complete the action on the overlapping areas where it is involved. The constraints in (21) ensure that each actor use one among its power levels, and that at least one actor act on each overlapping area, respectively. Constraint (22) limits the overall action completion time $\left(\sum_{a=1}^{N} \sum_{p=1}^{L} \frac{x_{a,p}^{(m)}}{T_{a,p}^{(m)}} \right)^{-1}$ to be smaller than the action completion bound, for each overlapping area. The constraints in (23) define the relation between the $x_{a,p}^{(m)}$ and h_a variables, while constraint (24) imposes that each actor act only on areas in its action range.

5. ACTOR-ACTOR COORDINATION: LOCALIZED AUCTION PROTOCOL

In this section, we propose a distributed solution to the actor-actor coordination problem stated in Section 4. In particular, our solution is inspired by the behavior of agents in a *market economy*, where the allocation of resources occurs as a result of interactions between buyers and sellers [25][26]. Our approach is based on a *real-time auction protocol* that describes the behavior of actors participating in transactions as buyers/sellers. The objective of the auction is to select the best set of actors to perform the action on each overlapping area. Thus, overlapping areas are *items* that are traded by the actors. The actors can assume the following roles:
- *Seller.* Is the actor responsible for a portion of event area, i.e., the actor that receives event features for that area. It corresponds to a collector.
- *Auctioneer.* Is the actor in charge of conducting the auction on a particular overlapping area. An auctioneer is selected for each overlapping area by the collector/seller responsible for that area.
- *Buyer.* The actors that can act on a particular overlapping area are referred to as buyers for that area.

A localized auction takes place in each overlapping area. The *bid* of each actor participating in the auction consists of a power level and of the corresponding *action completion time* (i.e., the time needed by that actor to complete the action on the whole area) defined in (14) and (15), respectively, as well as the available energy of the actor. The objective is to maximize the total *revenue* of the team, where the team is constituted by the actors participating in the auction, and the revenue depends on the *residual energy* (i.e., E_{Avg}^{Res}, in Section 4). *Multiple localized auctions* take place in parallel under the responsibility of different auctioneers. This is preferable to one single auction conducted by the collector for several reasons: i) it causes lower signaling overhead, since the auction messages are exchanged between the auctioneer and the buyers for that overlapping area, which are close to the auctioneer. This way, the message exchange is localized since the messages are not forwarded to the collector, which may be far from the overlapping area; ii) the auction process workload is shared among a higher number of actors, since the number of auctioneers is in general higher than the number of collectors.

When seller c (the collector) receives the event features from the sensors, it decides whether an action needs to be performed on the area it is responsible for. It computes all the non-overlapping and overlapping areas, since it knows the position and action ranges of the other actors. The coordination problem arises for the overlapping areas where more than one actor can act, while for the non-overlapping areas the seller directly assigns the action task to the corresponding actor.

Seller c selects M_c *auctioneers*, one for each overlapping area, among the actors that can act on each of these areas. Let $s^{(m)} \in \mathcal{S}^A$ be the auctioneer selected by seller c to conduct the auction for the m^{th} overlapping area. This auctioneer is selected to be the closest actor to the center of the overlapping area. This way, since the auctioneer is close to each actor in the overlapping area, the energy spent for communication and the auction time is reduced. After selecting the auctioneer $s^{(m)}$, the seller c provides it with the area $\mathcal{A}_{c,ov}^{C,m}$ where the auction should take place, the *action completion bound* δ_c, and the *auction time bound* τ_c, which is the maximum allowed time for the auction.

The auctioneer determines the winners of the auction based on the bids it receives from the buyers. At the beginning of the auction, the auctioneer sends a `JOIN_AUCTION` message to all the buyers competing for the area. After a buyer a hears this announcement, it submits its available energy, E_a^{Av}, and L two-dimensional bids $\underline{b_a} = \{b_a^1, b_a^2, \ldots, b_a^L\}$, where $b_a^{(p)} = [P_{a,p}^{(m)}, T_{a,p}^{(m)}]$, $p = 1, 2, \ldots, L$, with $P_{a,p}^{(m)}$ and $T_{a,p}^{(m)}$ defined in (14) and (15), respectively. By means of these bids, the auctioneer determines the winners by calculating the optimal solution for the residual energy

maximization problem $\mathbf{P}_{Max}^{Res}$ defined in Section 4.2. However, in this case the problem is limited to the overlapping area the auctioneer is responsible for. This way, since the bids are submitted to the auctioneer only once, signaling overhead is reduced [27]. In microeconomic theory, our auction mechanism can be classified as a *single-round sealed-bid auction* [25], where each buyer submits its bids in one shot irrespective of the bids from other buyers. In other words, the auctioneer receives the bids from the buyers only once and then determines the winners (i.e., the optimal set of actors, and their power levels) accordingly.

6. PERFORMANCE EVALUATION

In this section, we present the performance evaluation of the proposed framework. In Section 6.1 we report the performance results for the sensor-actor coordination, while in Section 6.2 we discuss the performance results for the actor-actor coordination.

6.1 Sensor-Actor Coordination

The optimization problem presented in Section 2.3 was implemented with AMPL [28], and solved with CPLEX [29], which uses a branch and bound algorithm to solve mixed integer linear problems. The start-up, speed-up, and aggregation states, described in Section 3, were implemented in a C++ network-layer simulator and in the J-Sim Simulator [30], which implements the whole protocol stack of a sensor node, from physical to application layer, including 802.11 MAC, UDP transport and CBR traffic. All figures in this section report 95% confidence intervals. We considered three different simulation scenarios. In Scenario 1, the deployment area is circular with radius equal to $20m$. For each deployed sensor, the distance from the center of the area and the angle are uniformly distributed random variables. In Scenario 2, sensor nodes are randomly deployed in a square area of $25m$ x $25m$. The event area is circular, with varying radius ranging in $[2, 12]m$ in different simulations. The epicenter of the event area is randomly selected such that the event area completely falls into the terrain. Scenario 3 is similar to Scenario 2, but the side of the square area is $100m$. Four actors are randomly deployed in each scenario. As in [12], the simulation parameters for the energy model in Section 2.2 are chosen to be $E_{elec} = 50nJ/bit$, $\beta = 100pJ/bit/m^{\alpha}$, $\alpha = 4$. The transmission range of sensors is set to $10m$.

Since the global network behavior depends on several application-dependent parameters, we present performance evaluation results of particular network configurations that constitute upper and lower bounds on the achievable performance. Hence, in this section we refer to start-up configuration, speed-up configuration, and aggregation configuration, as those configurations where all nodes are in the start-up, speed-up, and aggregation state, respectively. This allows us to describe how the evolution among different states impacts the overall network performance and show the benefits of the proposed solution without making it dependent on the choice of parameters that govern the transitions among different states. Evaluating and tuning the dependence of network dynamics on the parameters that govern the transitions of nodes among the various states is left for future work.

Figure 4 shows a comparison between the optimal solution to the event-driven clustering problem described in Section 2 and the energy consumption in the start-up, speed-up, and aggregation configuration in Section 3, respectively, with varying event ranges. The overall network cost (energy needed to transmit one bit from each source to the actors) is reported in the figure. Noticeably, the optimal solution is almost independent of the event range. This happens because of two contrasting phenomena. The number of sources increases when the event range increases, leading to a potentially higher energy consumption; conversely, since more nodes are involved, aggregation can be increasingly leveraged. These two trends compensate each other leading to a flat curve. Conversely, the energy consumption in the start-up and speed-up configurations highly increases with the event range. As also shown in Fig. 4, this can be partially compensated by the benefits of the aggregation state. In particular, an aggregation configuration can be reached both from a start-up configuration and from a speed-up configuration. An aggregation configuration reached from a start-up configuration leads to an almost-optimal energy consumption, where as by reaching the aggregation configuration from a speed-up configuration, the energy consumption can still be decreased consistently, but not as much as in the previous case. The structure of the da-trees after the start-up/speed-up configuration somehow constrains an aggregation process based on simple logic and minimal interaction among sensors. Hence, Fig. 4 motivates the design of our three-state distributed protocol for sensor-actor coordination. In fact, the distributed solution described in Section 3 adapts the structure of the da-trees to reach an energy configuration that is between the speed-up line and the aggregation from start-up line in Fig. 4. Depending on the required latency bound and reliability threshold, after a transient start-up configuration, a certain number of sensors will enter the speed-up/aggregation state to reach a minimum energy configuration, given the required reliability. When a higher reliability is required, the network will move towards a higher energy/lower delay configuration, while when the required reliability is guaranteed with some margin, the network will move towards a lower energy configuration.

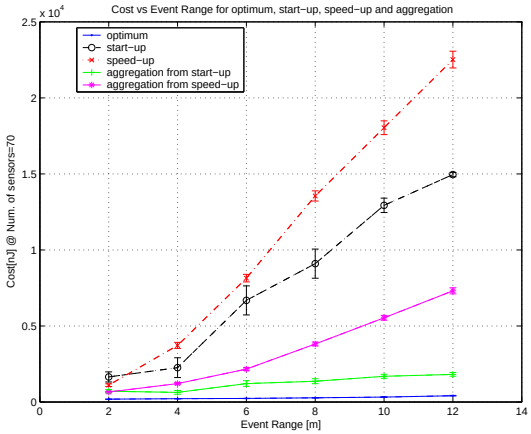

Figure 4: Scenario 1. Comparison of optimal solution, speed-up, start-up, and aggregation configuration with 70 nodes.

In Figures 5 and 6 we plot the average energy consumption versus the number of sensors and with different event ranges, for the start-up, and aggregation configurations in Scenario 2. As can be seen, the energy expenditure of the aggregation configuration is two orders of magnitude lower than in the start-up configuration (tens of thousands of nJ versus hundreds of nJ). This clearly shows the benefits of the aggregation configuration. As can be seen in Fig. 6, the energy expenditure increases less than the number of sensors, i.e., by doubling the number of sensors the energy expenditure is less than doubled. Figure 7 reports the overall energy consumption for the speed-up configuration. Interestingly enough, not only is the energy consumption of the speed-up configuration around one order of magnitude higher than in the start-up configuration; also, as already seen in Fig. 4, when the aggregation configuration is

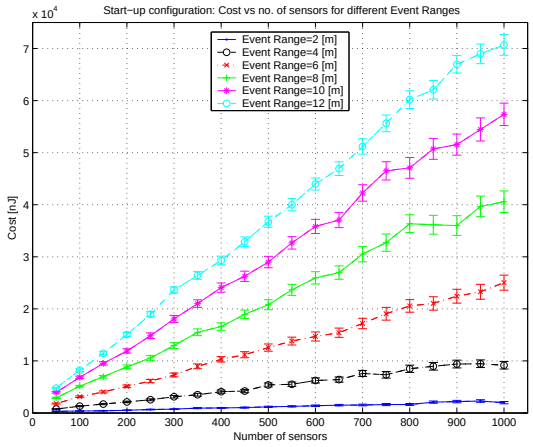

Figure 5: Scenario 2. Start-up configuration: Energy consumption vs. Number of sensors for different Event Ranges.

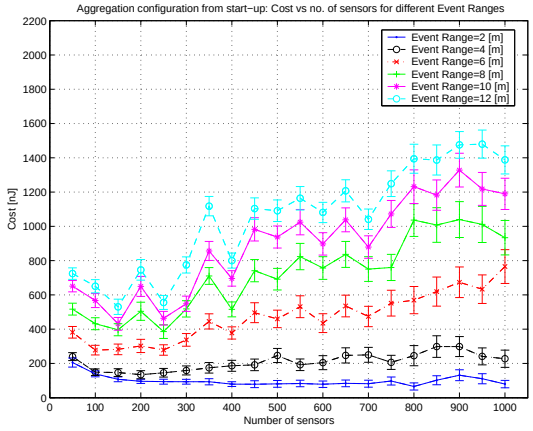

Figure 6: Scenario 2. Aggregation configuration: Energy consumption vs. Number of sensors for different Event Ranges.

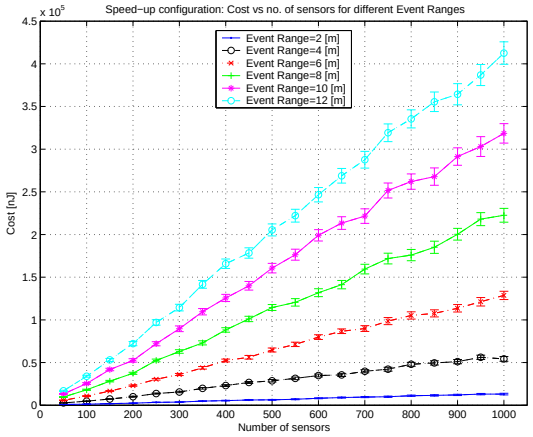

Figure 7: Scenario 2. Speed-up configuration: Energy consumption vs. Number of sensors for different Event Ranges.

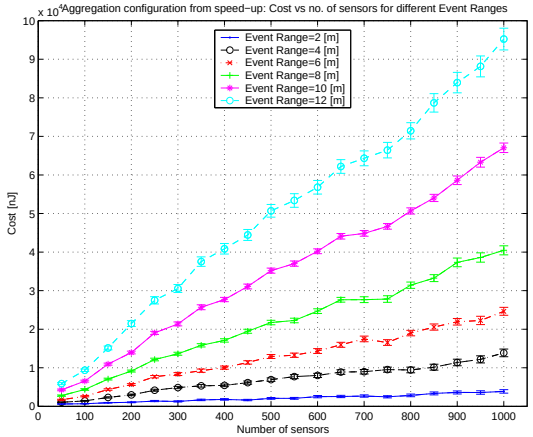

Figure 8: Scenario 2. Aggregation configuration reached from speed-up configuration: Energy consumption vs. Number of sensors for different Event Ranges.

reached from a speed-up configuration, the network converges to a less energy-efficient configuration with respect to the case when the aggregation configuration is reached directly from the start-up configuration. This is confirmed by Fig. 8, which shows that the order of magnitude of the energy consumption is 10^4 nJ for an aggregation configuration reached from a speed-up configuration. Conversely, as shown in Fig. 9, in Scenario 2 the average number of hops of each source-actor pair is reduced from around 5 hops for the start-up configuration to less than 2 hops in the speed-up configuration. This explains the rationale for the design of the distributed algorithm in Section 3. The speed-up configuration leads to paths with lower delay (lower number of hops and straight towards the destination); however, since this is paid with a higher energy consumption, the speed-up mechanism should be used only when strictly necessary to provide the required reliability.

Figure 10 shows the overall energy consumption for the start-up, speed-up, and aggregation configurations in Scenario 3, with 1000 nodes. Although the speed-up configuration can be seen to lead to a higher energy consumption, the energy consumptions of the start-up and speed-up configurations are in the same order of magnitude, i.e., the behavior of the speed-up configuration is sim-

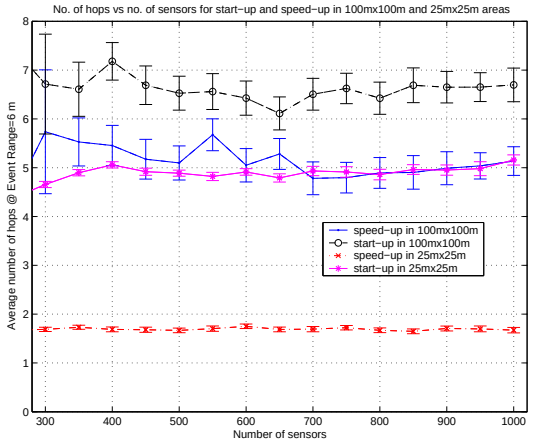

Figure 9: Scenario 2-3. Average number of hops for start-up and speed-up configurations.

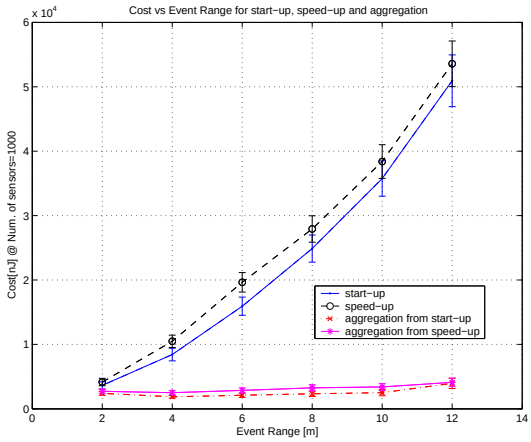

Figure 10: Scenario 3. Comparison of energy consumptions for start-up, speed-up, and aggregation configurations with 1000 sensors.

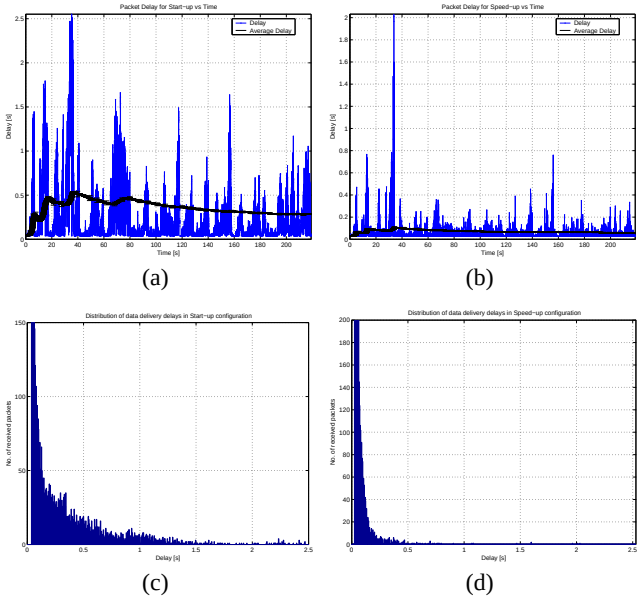

Figure 11: Scenario 3: Delays (a-b) and distribution of delays (c-d) for start-up and speed-up configurations, 400 sensors, event range = $12m$, sources generating 2 packets/s.

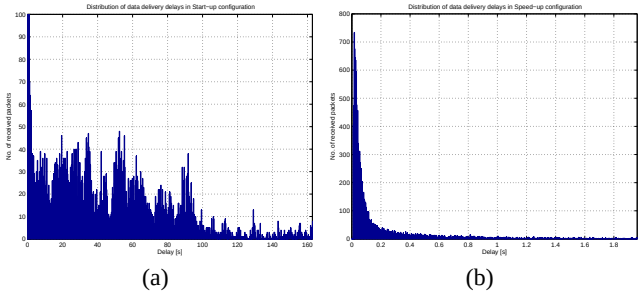

Figure 12: Scenario 3: Distribution of delays (a-b) for start-up and speed-up configurations, 400 sensors, event range = $12m$, sources generating 5 packets/s.

ilar to that of the start-up configuration. This happens when the transmission range of the nodes is short with respect to the distance between sensors and actors. In this case, a node in the start-up state tends to select the closest node to the destination (as it would do in the speed-up configuration). In fact, when the distance between the sensor and the actor is much larger than the transmission range, the second term in the sum of the two-hop rule accounts for most of the energy expenditure (see Section 3.2). In fact, the distance between the sensor and any of its neighbors is short as compared to the distance between the neighbor and the actor. Thus, the neighbor is selected so as to minimize the second term, which, as can be easily demonstrated, results in selecting the neighbor that is closest to the destination (as also a node in the speed-up state does). Nevertheless, Fig. 9 shows that the speed-up configuration still outperforms the start-up configuration in terms of number of hops, while this is achieved with a limited additional energy expenditure. More importantly, this is reflected in the distribution of packet delays, where the speed-up configuration always leads to lower delays than the start-up configuration. Figures 11 and 12 show a comparison of packet delays from sensor to actor in Scenario 3, with 400 nodes, between the speed-up and the start-up configuration when the event range is set to $12m$, which corresponds to 20 sources in average. In Figures 11(a)-11(b), sources generate 2 packets per second, each 32 bytes long, for 200 seconds of simulation. The variability of packet delays is reported as well as the cumulative average with time. In the start-up configuration, delays are shown to be high in the transient phase at the beginning of the simulation. The average delay (thick line in the figure) converges to a value around $0.3s$. In the speed-up configuration delays are much smaller, and their average is below $0.1s$. Figures 11(c)-11(d) show the distribution of delays in the same scenario. In the speed-up configuration (Fig. 11(d)), the delay is below 0.5 s for almost 100% of the packets, while in the start-up configuration (Fig. 11(c)) the variability of delays is much higher and their value can be as high as $2.5s$.

Figures 12(a) and 12(b) refer to the same scenario, where each source generates 5 packets per second, for the start-up and speed-up configurations, respectively. Noticeably, while in the start-up configuration the network is congested (Fig. 12(a)), leading to extremely high values for the delays, this does not happen in the speed-up configuration, where the delays are shown to be almost always within $1s$ (Fig. 12(b)). Note that, since the latency bound is application dependent, in these simulations we do not drop packets at intermediate nodes.

6.2 Actor-Actor Coordination

In this section, we discuss some performance results of the actor-actor coordination problem defined in Section 4. The model of the MINLP problem was implemented in AMPL and solved with the MINLP solver available through the NEOS Optimization Server [17]. In Fig. 13, we compare the average residual energy with three different solution approaches, namely, the optimal (Section 4.2), *1-actor* and *localized auction* (see Section 5). In the optimal solution, the best set of actors is chosen so that the average residual energy of the actors involved is maximized, while guaranteeing that the action is completed before the action completion time. In the 1-actor heuristic, the action is performed by one actor only for each overlapping area, i.e., the actor with the highest available energy. In the localized auction each overlapping area is taken care of by an auctioneer that splits it among the actors based on their bids, as discussed in Section 5. In the experiments performed, we concentrate on a scenario with three overlapping areas.

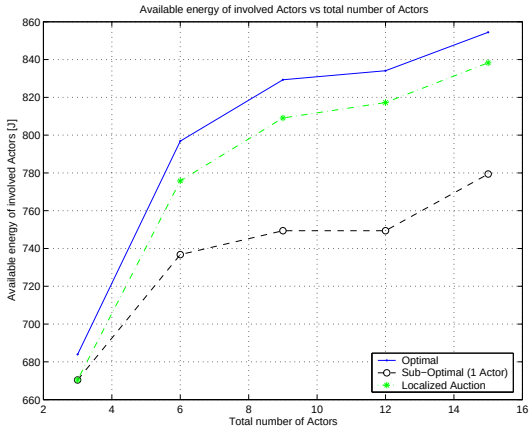

Figure 13: Available energy of the involved actors: optimal solution, localized auction, and 1-actor heuristic.

For the parameters defined in Section 4 we assumed the following values: $A_{c,ov}^{C,1} = 50m^2$, $A_{c,ov}^{C,2} = 100m^2$, $A_{c,ov}^{C,3} = 150m^2$, $P_a^{Max} = 100W$, $L = 5$, $K/\eta_a = 1W^{\gamma} \cdot s/m^2$, $\delta_c = 10s$, $\gamma = 0.8$. The values of the initial available energy of the actors are random variables uniformly distributed between $800J$ and $1000J$. As shown in Fig. 13, the localized auction mechanism leads to near-optimal residual energy, as each auctioneer calculates the optimal solution separately for its overlapping area. However, this greatly simplifies the problem and can be achieved with local communications among actors.

7. CONCLUSIONS

We presented a coordination framework for Wireless Sensor and Actor Networks (WSANs), and discussed the sensor-actor and actor-actor coordination problems. We developed an optimal solution for the sensor-actor coordination based on event-driven clustering, and formulated it as an *Integer Linear Program (ILP)*. We also proposed a distributed solution that includes an adaptive mechanism to trade off energy consumption for delay when the event data has to be delivered to the actors within pre-determined latency bounds. For the actor-actor coordination, an optimization model was defined for a class of coordination problems in which the area to be acted upon is optimally split among different actors. The problem was formulated as a mixed integer non-linear problem, and an auction-based localized solution of the problem was presented. Future work will be focused on extending the performance analysis of the proposed solutions, e.g., parameter fine tuning, and on generalizing the framework to capture different applications scenarios, including those requiring mobile actors.

Acknowledgments

This work was supported by the NSF under contract ECS-0428329.

8. REFERENCES

[1] I. F. Akyildiz and I. H. Kasimoglu, "Wireless sensor and actor networks: Research challenges," *Ad Hoc Networks (Elsevier)*, vol. 2, no. 4, pp. 351–367, October 2004.

[2] T. He, J. Stankovic, C. Lu, and T. Abdelzaher, "SPEED: A real-time routing protocol for sensor networks," in *Proceedings of IEEE ICDCS*, Providence, RI, USA, May 2003, pp. 46–55.

[3] C. Lu, B. M. Blum, T. F. Abdelzaher, J. A. Stankovic, and T. He, "RAP: a real-time communication architecture for large-scale wireless sensor networks," in *Proceedings of IEEE RTAS*, San Jose, CA, USA, September 2002.

[4] E. Felemban, C.-G. Lee, E. Ekici, R. Boder, and S. Vural, "Probabilistic QoS Guarantee in Reliability and Timeliness Domains in Wireless Sensor Networks," in *Proceedings of IEEE INFOCOM 2005*, Miami, FL, USA, Mar. 2005.

[5] J. A. Stankovic, T. F. Abdelzaher, C. Lu, L. Sha, and J. Hou, "Real-time communication and coordination in embedded sensor networks," *Proceedings of the IEEE*, vol. 91, no. 7, pp. 1002–1022, 2003.

[6] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Computer Networks (Elsevier)*, vol. 38, no. 4, pp. 393–422, Mar. 2002.

[7] Y. Yu, B. Krishnamachari, and V. K. Prasanna, "Energy-latency tradeoffs for data gathering in wireless sensor networks," in *Proceedings of IEEE INFOCOM 2004*, Hong Kong S.A.R., P.R. China, 2004.

[8] J. Li, J. Jannotti, D. D. Couto, D. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing," in *Proceedings of ACM/IEEE MobiCom 2000*, Boston, Massachussets, 2000, pp. 120–30.

[9] R. Jain, A. Puri, and R. Sengupta, "Geographical routing using partial information for wireless ad hoc networks," *IEEE Personal Communications*, pp. 48–57, February 2001.

[10] J. Hightower and Borriello, "Location systems for ubiquitous computing," *IEEE Computer*, vol. 34, no. 8, pp. 57–66, Aug. 2001.

[11] F. Kuhn, R. Wattenhofer, and A. Zollinger, "Worst-Case Optimal and Average-Case Efficient Geometric Ad-Hoc Routing," in *Proceedings of ACM MobiHoc 2003*, Annapolis, MD, USA, June 2003.

[12] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, Oct. 2002.

[13] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," in *Proceedings of IEEE INFOCOM 2004*, Hong Kong S.A.R., P.R. China, March 2004.

[14] V. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, and N. Shroff, "A minimum cost heterogeneous sensor network with a lifetime constraint," *IEEE Transactions on Mobile Computing*, vol. 4, no. 1, pp. 4–15, Jan. 2005.

[15] F. Kuhn, T. Moscibroda, and R. Wattenhofer, "Initializing newly deployed ad hoc and sensor networks," in *Proceedings of ACM MobiCom 2004*, Philadelphia, PA, September 2004.

[16] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, New Jersey: Prentice Hall, February 1993.

[17] J. Czyzyk, M. Mesnier, and J. Moré, "The NEOS server," *IEEE Journal on Computational Science and Engineering*, vol. 5, pp. 68–75, 1998.

[18] B. Krishnamachari, D. Estrin, and S. Wicker, "The impact of data aggregation in wireless sensor networks," in *Proceedings of IEEE DEBS 2002*, Vienna, Austria, July 2002.

[19] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Communications of the ACM*, vol. 43, pp. 51–58, May 2000.

[20] M. R. Garey and D. S. Johnson, *Computer and Intractability*. New York, NY: W. H. Freeman and Co., 1979.

[21] T. Melodia, D. Pompili, and I. F. Akyildiz, "On the Interdependence of Distributed Topology Control and Geographical Routing in Ad Hoc and Sensor Networks," *IEEE Journal of Selected Areas in Communications*, vol. 23, no. 3, pp. 520–532, March 2005.

[22] B. Sundararaman, U. Buy, and A. Kshemkalyani, "Clock synchronization for wireless sensor networks: a survey," *Ad Hoc Networks (Elsevier)*, vol. 3, no. 3, pp. 281–323, May 2005.

[23] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks," *ACM Wireless Networks*, vol. 7, no. 6, pp. 609–616, Nov. 2001.

[24] G. Finn, "Routing and addressing problems in large metropolitan-scale internetworks," ISI res. rep ISU/RR- 87-180, Tech. Rep., March 1987.

[25] R. P. McAfee and J. McMillan, "Auctions and bidding," *Journal of Economic Literature*, vol. 25, no. 2, pp. 699–738, June 1987.

[26] B. P. Gerkey and M. J. Mataric, "Sold!: Auction methods for multirobot coordination," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 758–768, October 2002.

[27] P. Maillé and B. Tuffin, "Multi-bid auctions for bandwidth allocation in communication networks," in *Proceedings of IEEE INFOCOM 2004*, Hong Kong S.A.R., P.R. China, March 2004.

[28] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press / Brooks/Cole Publishing Company, 2002.

[29] CPLEX solver. [Online]. Available: http://www.cplex.com

[30] The J-Sim Simulator. [Online]. Available: http://www.j-sim.org/