Final Report for NSF

Award Number: 0428329

Correlation-Based Collaborative Advanced Communication Protocols for Wireless Sensor Actor Networks (WSAN) - September 2007

Ian F. Akyildiz
Raghupathy Sivakumar

School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332

Tel: (404)-894-5141; Fax: (404)-894-7883
E-Mail: {ian, siva}@ece.gatech.edu

Contents

Summary

Wireless Sensor and Actor Networks (WSANs) [5] are distributed wireless systems of heterogeneous devices referred to as *sensors* and *actors*. Sensors are low-cost, low-power, multi-functional devices that communicate untethered in short distances. Actors collect and process sensor data and consequently perform actions on the environment. In most applications, actors are resource rich devices equipped with high processing capabilities, high transmission power, and long battery life. In WSANs, the collaborative operation of the sensors enables the distributed sensing of a physical phenomenon. After sensors detect an event that is occurring in the environment, the event data is distributively processed and transmitted to the actors, which gather, process, and eventually reconstruct the event data. The process of establishing data paths between sensors and actors is referred to as sensor-actor coordination. Once the event has been detected, actors coordinate to reconstruct it, to estimate its characteristics, and make a collaborative decision on how to perform the action. This process is referred to as actor-actor coordination. As a result, the operation of a WSAN can be thought of as an event-sensing, communication, decision, and acting loop. In this report, we summarize the research efforts on Wireless Sensor and Actor Networks in the framework of the NSF-sponsored project "Correlation-Based Collaborative Advanced Communication Protocols for Wireless Sensor Actor Networks" at Georgia Tech.

1 Introduction

The convergence of communication and computation with signal processing and several branches of control theory such as robotics and artificial intelligence is enabling distributed systems of embedded devices that sense, interact, and control the physical environment. Wireless Sensor and Actor[1] Networks (WSANs) [5] are distributed wireless systems of heterogeneous devices referred to as *sensors* and *actors*. Sensors are low-cost, low-power, multi-functional devices that communicate untethered in short distances. Actors collect and process sensor data and consequently perform actions on the environment. In most applications, actors are resource rich devices equipped with high processing capabilities, high transmission power, and long battery life.

Our research on WSANs presents several overlaps with other research disciplines, and aims at complementing and integrating them. For example, *Distributed Robotics* [10] has been a hot research topic since the mid 90's. In distributed robotics, a task is not completed by a single robot but by a *team of collaborating robots*. Information about the surrounding environment is usually gathered by *onboard sensors*, and team members exchange sensor information to move or perform actions (e.g., collaborate to manipulate heavy objects). As opposed to a single robot, a team of robots can perceive the environment from *multiple disparate viewpoints*. In WSANs, the ability of the actors to perceive the environment can be pushed one step further: a dense spatio-temporal sampling of the environment, provided by a pre-deployed sensor network, can be exploited by the whole team of actors, thus increasing the ability of the team to accurately interact with the physical environment. This also enables a new design perspective, as it makes possible to build simpler, less expensive robots easier to maintain and debug that can accomplish the goal reliably through cooperation with the sensor network.

In WSANs, the collaborative operation of sensors enables the *distributed sensing* of a physical phenomenon. After sensors detect an event that is occurring in the environment, the event data is distributively processed and transmitted to the actors, which gather, process, and eventually reconstruct the event data. The process of establishing data paths between sensors and actors is referred to as *sensor-actor coordination* [5]. Once the event has been detected, the actors coordinate to reconstruct it, to estimate its characteristics, and make a collaborative decision on how to perform the action. This process is referred to as *actor-actor coordination* [5]. As a result, the operation of a WSAN can be thought of as an event-sensing, communication, decision, and acting loop.

Several applications for WSANs are concerned with *enhancing and complementing existing sensor network applications*. In these applications, the performed actions serve the purpose of enhancing the operation of the sensor network by enabling or extending its monitoring capability. For example, mobile actors can accurately deploy sensors [61], enable adaptive sampling of the environment [37], pick up data from the sensors when in close range, buffer it, and drop off the data to wired access points [46], or perform energy harvesting [40].

Conversely, we are concerned with new applications where actors are part of the network and perform

[1] It may be worth specifying how the term *actor* differs from the more conventional notion of *actuator*. From our perspective an actor, besides being able to act on the environment by means of several actuators, is also a *single network entity* that performs networking-related functionalities, i.e., receive, transmit, and relay data. For example, the mobility of a robot may be enabled by several motors and servo-mechanisms (actuators). However, from a networking perspective, the robot constitutes a single entity, which we refer to as actor.

actions based on the information gathered by sensors. We envision that WSANs will be an integral part of systems such as battlefield surveillance, nuclear, biological or chemical attack detection, home automation, and environmental monitoring [5]. For example, in fire detection applications, sensors can relay the exact origin and intensity of the fire to water sprinkler actors that will extinguish the fire before it spreads. Moreover, sensors can detect plumes, i.e., visible or measurable discharges of contaminants in water or in the air, and actors can reactively take countermeasures. Similarly, motion, acoustic, or light sensors in a building can detect the presence of intruders and command cameras or other instrumentations to track them. Alternatively, mobile actors can be moved to the area where the intruder has been detected to get high resolution images, prompt or block the intruder. Another promising application for WSANs are so-called Pursuit Evasion Games (PEGs) [43], where a group of pursuer actors is required to detect, chase and capture a group of evaders in minimum time with the aid of a sensor network. As a last example, in earthquake scenarios sensors can help locate survivors and guide mobile actors performing rescue operations.

In Section 2, we propose a framework for communication and coordination problems with static WSANs. The concepts of *sensor-actor coordination* and *actor-actor coordination* are introduced, and centralized optimal solutions and distributed heuristics proposed. However, many challenging applications require support for mobile actors, which is not provided by the algorithms in Section 2. Hence, in Section 3, we extend our previous work in several directions. First, we introduce a hybrid location management scheme to handle the mobility of actors with minimal energy expenditure for the sensors, develop an integrated routing/physical layer scheme for sensor-actor communication based on geographical routing, which is suited for mobile WSANs. In case of high traffic, we introduce a new network congestion control mechanism at the network layer that forces multiple actors to share the traffic generated in the event area. A new model for actor-actor coordination is introduced that enables coordinating motion and action of the participating actors based on the characteristics of multiple, concurrent events.

To address the need for real-time and reliable data transport, in Section 4, we introduce a real-time and reliable transport $(RT)^2$ protocol for WSANs. $(RT)^2$ is a novel transport solution that seeks to *achieve reliable and timely event detection with minimum possible energy consumption and no congestion.* It enables the applications to perform right actions timely by exploiting both the correlation and the collaborative nature of WSANs.

Then, the evolution from WSNs, which can be thought of to perform only *read* operations, to WSANs, which can perform both *read and write* operations, introduces unique and new challenges that need to be addressed. In this context, we study two problems pertaining to correctness of operation, and propose distributed primitives to address the problem. First, in Section 5 we consider the problem of causal and correct execution of queries and commands in the distributed WSAN, which we refer to as hazard avoidance [57, 56, 54, 53]. Second, in Section 6 we consider the problem of performing actions only to the appropriate level in the distributed environment, referred to as mutual exclusion [55]. We will describe these two problems in WSANs, and the distributed primitives to address these problems.

Finally, in Section 7 we investigate the energy efficiency of the correlation aware aggregation process through comprehensive quantitative analysis and leverage the insights gained to design a distributed and highly

efficient aggregation solution. We specifically explore how the improvement in energy efficiency is impacted by network conditions, defined by several parameters including the node density, source density, the physical distribution of sources, the correlation degree, and the delay bound. We present observations from the simulation results, and draw inferences on the trade-offs involved in achieving energy efficiency.

2 A Coordination Framework for Wireless Sensor and Actor Networks

In WSANs, the collaborative operation of the sensors enables the *distributed sensing* of a physical phenomenon. After sensors detect an event that is occurring in the environment, the event data is distributively processed and transmitted to the actors, which gather, process, and eventually reconstruct the event data. The process of establishing data paths between sensors and actors is referred to as *sensor-actor coordination* [5]. Once the event has been detected, the actors coordinate to reconstruct it, to estimate its characteristics, and make a collaborative decision on how to perform the action. This process is referred to as *actor-actor coordination* [5]. As a result, the operation of a WSAN can be thought of as an event-sensing, communication, decision, and acting loop.

Some recent papers [23][17] have considered the issue of real-time communication in sensor networks. However, as discussed in [48], there are still many open research challenges 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, i.e., define how actors and sensors communicate. As far as scalability is concerned, it has been pointed out [34][28] that the routing protocols that do not use geographical location information do not scale well. Therefore, in this section we study the *sensor-actor coordination* based on a geographical routing paradigm. Several solutions propose to guarantee scalability and energy efficiency based on partitioning the sensor network into different clusters [62][33]. 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.

In this section, we propose to base the sensor-actor coordination on an *event-driven clustering* paradigm, where cluster formation is triggered by an event and clusters are created *on-the-fly* to optimally react to the event. In our approach, sensors detecting an event coordinate 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. The resulting architecture is shown in Fig. 1. In addition, we introduce a model for actor-actor coordination, whose objective is to optimally allocate tasks to the different actors to collaboratively achieve a global goal. 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.

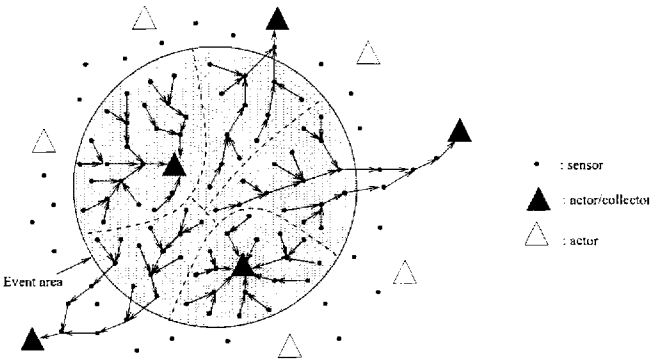


Figure 1: Event-driven clustering with multiple actors.

2.1 Sensor-Actor Coordination

2.1.1 Problem Formulation

As discussed in the previous section, sensor-actor communications may have real-time requirements. Hence, we introduce a novel notion of reliability that accounts for the percentage of packets generated by the sensors in the event area 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, and is calculated at the network layer.

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.*

Definition 2 *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 B is said to be* unexpired *and thus,* reliable.

Definition 3 *The* event reliability r *is the ratio of* reliable *data packets over all the packets generated in a decision interval* [2]. *The* event reliability threshold r_{th} *is the minimum* event reliability *required by the application.*

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

[2]Whenever one or more packets are dropped by an intermediate sensor, the actor is notified about the lost packet(s) in the header of the next data packet, so that the packet loss can be taken into account in the computation of the reliability.

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). 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. The 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. The optimal strategy for event-driven clustering is formulated as an *Integer Linear Program* (ILP) [3].

2.1.2 Distributed Protocol

In this section, we introduce a scalable and distributed protocol to address 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, to provide the required reliability r_{th} with minimum energy expenditure. We refer to the protocol as *Distributed Event-driven Clustering and Routing* (DECR) protocol. We seek a solution that results in a good compromise between the energy efficiency of the established da-trees, and the amount of topology information needed by each sensor to take a routing decision. Conversely, complying with pre-determined delay bounds requires some form of end-to-end feedback. Instead of requiring feedback information for each individual source, which would cause unacceptable overhead, we rely on collective feedback from the receiving actors. Each actor advertises the observed reliability. Based on this, 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 event reliability r above the required threshold r_{th} and minimize the energy consumption. The reliability is controlled based on the idea of adjusting the delays, by modifying the average end-to-end path length. While modifying the energy consumption in an ad hoc network by changing the transmitted power is common practice, the proposed protocol can be also seen as a mechanism to adjust the end-to-end delay based on transmission power control. To the best of our knowledge, this idea has not been thoroughly explored so far.

Each sensor alternates among four different states, namely *idle*, *start-up*, *speed-up*, and *aggregation state*. The main objective of these state transitions is to reduce the number of hops, which results in decreased delay, when the reliability requirement is violated; and to save energy when the reliability requirement is met. This is achieved by probabilistically modifying the behavior of the sensor nodes at the routing layer, i.e., inducing them to select their next hop so as to increase the delay and reduce the energy consumption when the reliability is high, and viceversa reducing the delay at the expense of energy consumption when the reliability is low. In general, 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 generated packets 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.

2.2 Actor-Actor Coordination

2.2.1 Problem Formulation

The objective of the actor-actor coordination is to select the best actor(s) to perform appropriate action on the event area. Actor-actor coordination presents several analogies with the so-called *Multi-Robot Task Allocation* (MRTA) problem encountered in robotics. In fact, a fundamental questions faced when designing cooperative multi-robot systems is "Which robot should execute which task in order to cooperatively achieve the global goal?" [19]. At the end of the sensor-actor coordination phase, 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*). 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 the action. 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*. 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*. 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.

[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.

2.2.2 Localized Auction Protocol

In this section, we propose a distributed solution to the actor-actor coordination problem. 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. It 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), 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*. *Multiple localized auctions* take place in parallel under the responsibility of different auctioneers.

2.3 Performance Evaluation

2.3.1 Sensor-actor Coordination

The optimization problem was implemented in AMPL [18], and solved with CPLEX [1]. The start-up, speed-up, and aggregation states, were implemented in a C++ simulator, which we used to evaluate the energy consumption, and in the J-Sim Simulator [2], which implements the whole protocol stack of a sensor node, from physical to application layer, including CSMA/CA MAC. All figures in this section report 95% confidence intervals. We considered 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. The simulation parameters for the energy model 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, here we present results related to 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 show the

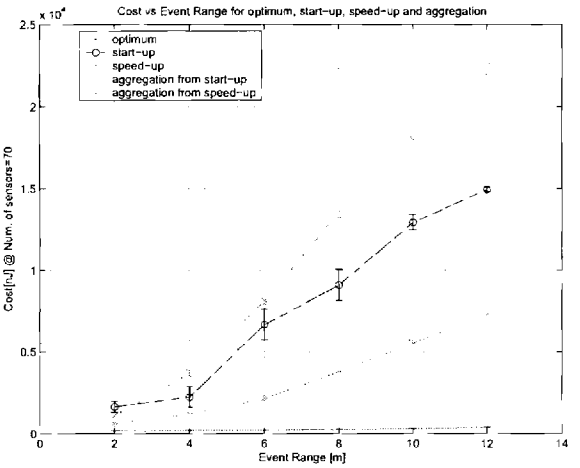
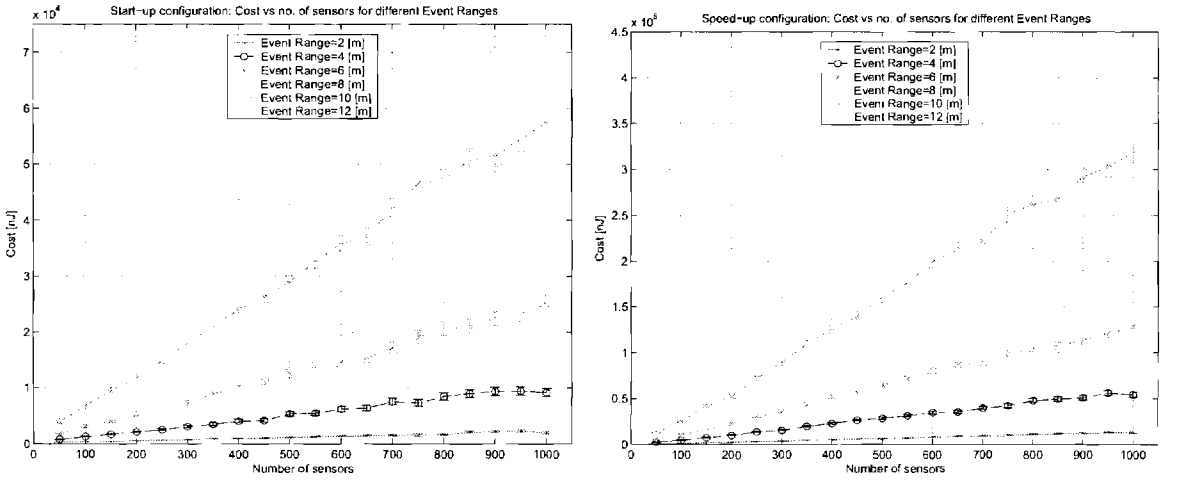


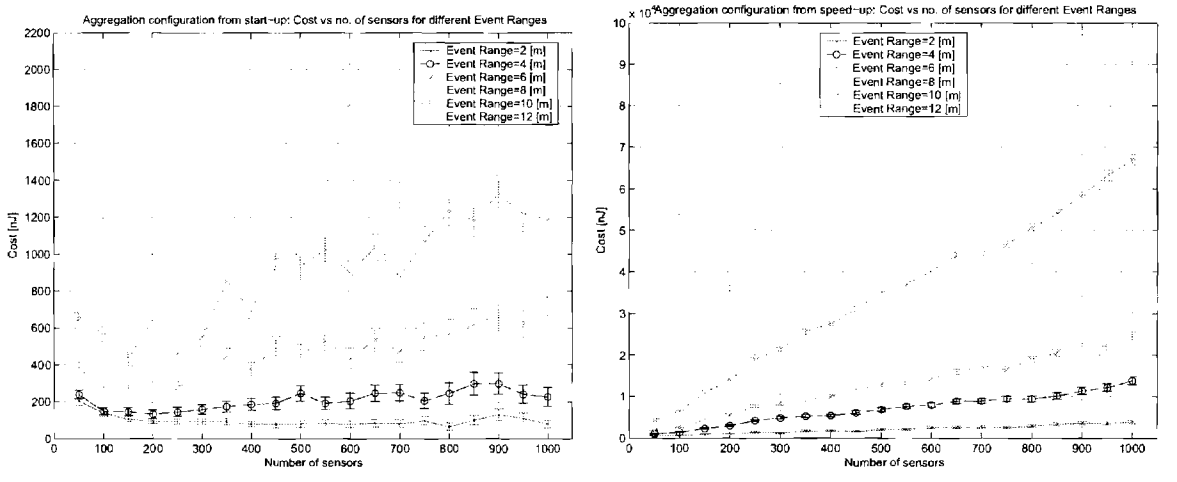
Figure 2: Scenario 1. Comparison of optimal solution, speed-up, start-up, and aggregation configuration with 70 nodes.

benefits of the proposed solution without depending on the choice of parameters that govern the transitions among states. Figure 2 shows a comparison between the optimal solution to the event-driven clustering problem, and the energy consumption in the start-up, speed-up, and aggregation configuration, with varying event ranges. The figure shows the overall network cost, i.e., the energy needed to transmit one bit from each source to the actors. Noticeably, the optimal solution is almost independent of the event range. This is due to 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. In Figs. 3(a) and 4(a) we plot the average energy consumption versus the number of sensor, with different event ranges, for the start-up and aggregation configurations in Scenario 2. The energy expenditure of the aggregation configuration is two orders of magnitude lower than in the start-up configuration. As can be seen in Fig. 4(a), the energy expenditure increases sublinearly with the number of sensors. Figure 3(b) reports the overall energy consumption for the speed-up configuration. Interestingly, 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. 2, when the aggregation configuration is reached from a speed-up configuration, the network converges to a less energy-efficient configuration, compared to when the aggregation configuration is reached directly from the start-up configuration. This is confirmed by Fig. 4(b), 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. 5(a), 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. The speed-up configuration leads to paths with lower delay (lower number of hops); 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 5(b) shows the overall energy consumption for the start-up, speed-up, and aggregation configurations in



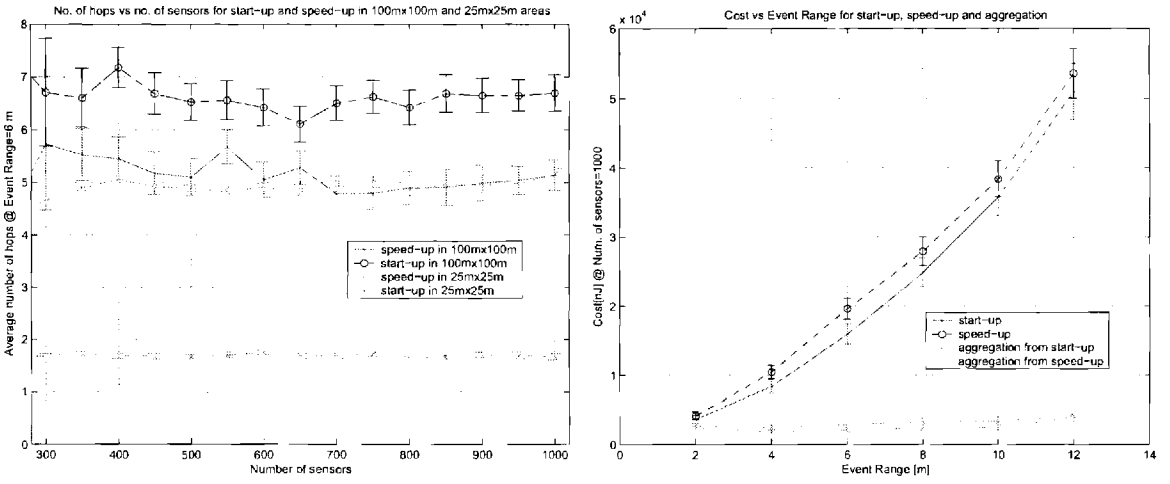
(a) (b)

Figure 3: Start-up (a) and speed-up (b) configuration: Energy consumption vs. Number of sensors for different Event Ranges.



(a) (b)

Figure 4: Scenario 2. Aggregation configuration reached from start-up (a) and speed-up (b) configuration: Energy consumption vs. Number of sensors for different Event Ranges.



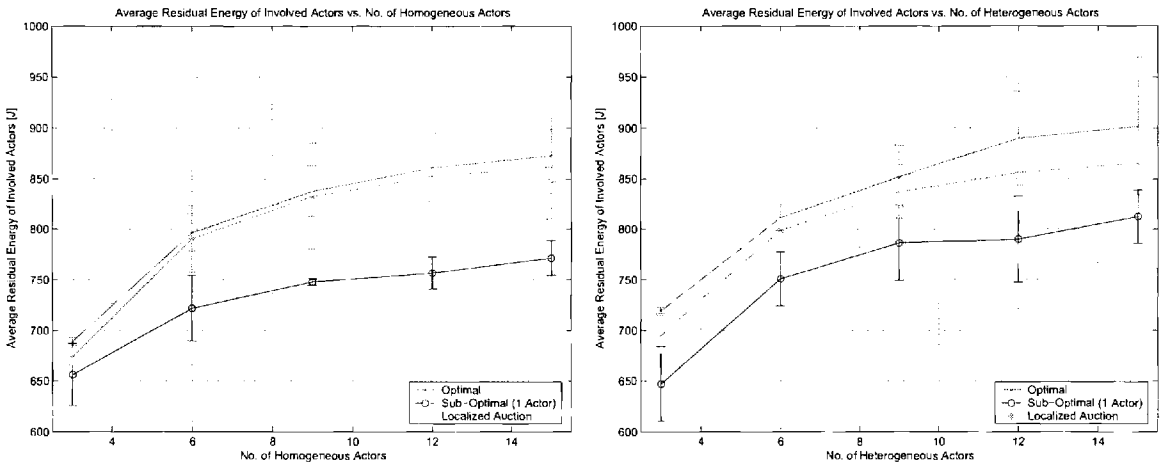
(a) (b)

Figure 5: (a) Scenario 2-3. Average number of hops for start-up and speed-up configurations, (b) Scenario 3. Comparison of energy consumptions for start-up, speed-up, and aggregation configurations with 1000 sensors.

Scenario 3, with 1000 nodes. By comparing Figs. 5(a) and 5(b), we can conclude that in this case 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. This is also reflected in the distribution of packet delays.

2.4 Actor-actor Coordination

In this section, we discuss some performance results of the actor-actor coordination problem. The model of the MINLP problem was implemented in AMPL and solved with the MINLP solver available through the NEOS Optimization Server [14]. In Figs. 6(a) and 6(b), we compare the average residual energy with three different solution approaches, namely, the *optimal*, *1-actor*, and *localized auction*. In the optimal solution, the best set of actors is chosen so that the average residual energy of the involved actors 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 residual energy after the completion of the action. In the localized auction each overlapping area is taken care of by an auctioneer that divides it among the actors based on their bids. In the experiments performed, we concentrate on two scenarios with three overlapping areas, one with homogeneous actors with $\gamma_a = 0.8$ (Fig. 6(a)), and one with heterogeneous actors half of which with $\gamma_a = 0.6$ (low-efficiency actors) and the other half with $\gamma_a = 0.9$ (high-efficiency actors) (Fig. 6(b)). For the remaining parameters, we assume the following values: $A_{c,ov}^{C,1} = 50m^2$, $A_{c,ov}^{C,2} = 100m^2$, $A_{c,ov}^{C,3} = 150m^2$, $P_a^{Max} = 100W$, $L = 5$, $K/\eta_a = 1W^{\gamma_a} \cdot s/m^2$, and $\delta = 10s$. The value of the initial available energy E_a^{Av} of an actor is a random variable uniformly distributed between $800J$ and $1000J$. As shown in both Figs. 6(a) and 6(b), 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. Moreover, in the heterogeneous scenario, the



(a) (b)

Figure 6: Average residual energy of the involved actors in the homogeneous (a) and heterogeneous case (b).

proposed localized solution effectively exploits the high-efficiency actors, thus reducing the dissipated energy to complete the action.

3 Handling Mobility in WSANs

As an abstraction of several application setups encountered in the above-mentioned applications, we refer to a scenario where sensors monitor a given terrain, and send samples of the event to the actors deployed on the terrain whenever an event occurs. Actors distributively reconstruct the event based on partial information available at different actors, estimate the event characteristics and identify an *action area*. Based on this, actors collaboratively decide on which actors should move to the action area and at which speed. The coordinated mobility of actors is thus triggered by the occurrence of events.

First, we introduce a hybrid location management scheme to handle the mobility of actors with minimal energy expenditure for the sensors. The proposed solution is tailored for WSAN applications and overcomes the drawbacks of previously proposed localization services [34][15]. Actors broadcast updates limiting their scope based on Voronoi diagrams, while sensors predict the movements of actors based on Kalman filtering of previously received updates. Our proposed scheme is shown to consistently reduce the energy consumption on sensors by avoiding over 75% of location updates with respect to existing location update algorithms.

The second contribution is the development of an integrated routing/physical layer scheme for sensor-actor communication based on geographical routing, which is suited for mobile WSANs. We derive a simple yet optimal forwarding rule based on geographic position in presence of Rayleigh fading channels. With respect to previously proposed geographic forwarding rules [45][41], our rule is optimal from the energy consumption standpoint. Furthermore, we show how to control the delay of the data-delivery process based on power control,

i.e., to trade optimal energy consumption for decreased delay in case of low or moderate traffic. In case of high traffic, we introduce a new network congestion control mechanism at the network layer that forces multiple actors to share the traffic generated in the event area. This is shown to reduce delay, packet drops and energy consumption even when traffic is sent to actors that are suboptimal from a network layer standpoint.

As a last contribution in our proposed system architecture, a new model for actor-actor coordination is introduced that enables coordinating motion and action of the participating actors based on the characteristics of multiple, concurrent events.

3.1 Location Management

Previous proposals have dealt with the development of scalable location services for tracking mobile nodes in distributed systems based on geographical routing. In [34], GLS was proposed, which is a hierarchical location service where each mobile node maintains its current location in a number of location servers distributed throughout the network. The location servers for each node are determined based on a hashing function in the node identifier space. In [15], the performance of GLS is compared to two other location services based on similar premises. In general, the objective of these mechanisms, which can be classified as rendezvous-based protocols [15], is to potentially allow each single device in the network to retrieve the location of any other node, based on queries and replies. Clearly, query-based mechanisms can introduce delays that may not be acceptable in delay-critical systems such as WSANs. Moreover, the extensive message exchange and complex server structures, often hierarchical, associated with these protocols, can be avoided given the characteristics of WSANs.

For this reason, we propose a proactive location management approach based on update messages sent by mobile actors to sensors. As discussed, in WSANs each actor is an equivalent recipient of information. Therefore, sensor-actor communications are localized, i.e., each sensor sends information to its closest actor. Hence, in the spatial domain, broadcasts can be limited based on Voronoi diagrams [7]. At the same time, actor movement is to some extent predictable, as it is driven by the actor-actor coordination procedures. Hence, in the temporal domain, location updates can be limited to *actor positions that cannot be predicted* at the sensor side. Location updates are triggered at the actors when the actual position of the actor is "far" from what can be predicted at the sensors based on past measurements. Therefore, actors that move following predictable trajectories, which is likely to be a common case in WSANs, will need to update their position much less frequently than actors that follow temporally uncorrelated trajectories.

3.1.1 Limiting Broadcasts in Space

As discussed, we propose to use Voronoi diagrams to limit the scope of actor-initiated location updates. The Voronoi diagram of a set of discrete sites partitions the plane into a set of convex polygons such that all points inside a polygon are closest to only one site.

A sensor s_i is said to be *dominated* by actor a_j if its location lies in the Voronoi cell of a_j. Every actor is responsible for location updates to sensors in its Voronoi cell, and regulates its power so as to limit interference beyond the farthest point in its Voronoi cell. Each sensor will thus expect to receive location updates from the actor it is dominated from. With respect to flooding, the energy consumption for location updates is drastically reduced. It can be shown that the worst-case energy consumption of a flooding scheme increases as a function order of $O(N_S^2 \cdot N_A)$, and most of the energy burden is on sensors. Conversely, if the actor is able to reach all sensors in its Voronoi cell in one hop, which may be true in many practical cases, the energy consumption increases as a function order of $O(N_S)$, and most of the energy burden is on actors.

3.1.2 Limiting Broadcasts in Time

The dynamic movement model for the i^{th} actor in two-dimensional coordinates can be described by a continuous time linear dynamical system.

The position observed by the actor at step k is related to the state by the *measurement equation*

$$\mathbf{z_i^k} = \mathbf{H}\mathbf{x_i^k} + \mathbf{C}\mathbf{v_i^k} \tag{1}$$

where $\mathbf{z_i^k} = [z_i^{k,x}, z_i^{k,y}]$ represents the *observed position* of the actor at step k, and where $\mathbf{H} = \begin{bmatrix} \mathbf{I} & 0 \end{bmatrix}$, $\mathbf{C} = \mathbf{B}$.

The variable $\mathbf{v_i^k} = [v_i^{k,x}, v_i^{k,y}]^T$ represents the *measurement noise*, expressed as two-dimensional samples of discrete time white Gaussian noise. Hence, $\mathbf{v_i^k} \sim \mathcal{N}(0, \mathbf{R})$, with $\mathbf{R} \geq 0$, where $\mathbf{R}$ is the covariance matrix of the process. The observed position of the actor $\mathbf{z_i^k}$ is thus the actual position of the actor affected by a measurement noise, which we represent as a Gaussian variable.

The Kalman filter [8] provides a computationally efficient set of recursive equations to estimate the state of such process, and can be proven to be the optimal filter in the minimum square sense. The joint use of Kalman filter at the sensor and actor sides enables reducing the number of necessary location updates. In fact, the filter is used to *estimate the position* at the actor based on measurements, which is a common practice in robotics, and to *predict* the position of the actors at the sensors, thus reducing the message exchange. The position of actor i can be estimated and predicted at the sensors in its Voronoi cell, based on the measurements $\mathbf{z_i^k}$ taken at the actor and broadcast by the actor. At step k, each sensor s in i's Voronoi cell updates the state (that represents position and velocity of the actor) based on the equations

$$\hat{\mathbf{x}}_{\mathbf{i,s}}^{\mathbf{k-}} = \mathbf{F}\hat{\mathbf{x}}_{\mathbf{i,s}}^{\mathbf{k-1}}, \quad \mathbf{P}_{\mathbf{i,s}}^{\mathbf{k-}} = \mathbf{F}\mathbf{P}_{\mathbf{i,s}}^{\mathbf{k-1}}\mathbf{F}^{\mathbf{T}} + \mathbf{Q}. \tag{2}$$

Note that the complexity of the above computations is very low as the size of the state space is only 4. Moreover, the processing cost for sensors is much lower than the communication cost. This is justified by [39], where the energy necessary to transmit 1 kbit is shown to be equivalent to the energy necessary to execute

300,000 processor instructions.

At each step k, actor i broadcasts the new measurement $\mathbf{z_i^k}$ if and only if a sensor s in its cell, which has received the previous updates, is not able to predict the position of the actor within a maximum error e_{max}, i.e., if $(\mathbf{z_i^k} - \mathbf{H\hat{x}_{i,s}^{k-}}) > e_{max}$. If sensor s does not receive a location update at step k, it assumes $\mathbf{z_i^k} = \mathbf{H\hat{x}_{(i,s)}^{k-}}$, i.e., the predicted position coincides with the actual new position of the actor. Based on this, it updates its estimate of the state for actor i.

3.2 Sensor-Actor Communication

In [36], we proposed a new notion of reliability that accounts for the percentage of packets generated by the sensors in the event area that are received within a pre-defined latency bound (which we refer to as *reliable packets*). The objective is to trade off energy consumption for latency when data has to be delivered within a given time bound B with a given reliability r_{th}. The solution presented in [36], based on similar premises, is however not suitable for mobile actors, as the convergence of the distributed protocol to an energy-efficient and latency compliant solution is too slow as compared to the dynamics encountered in networks with mobile actors. Therefore, when the traffic generated in the event area is low or moderate, we adjust the end-to-end delay by increasing the forwarding range with respect to the energy-efficient forwarding range, as described in Section 3.2.1. In case of congestion at a recipient actor, we re-route part of the traffic to another, less congested actor.

3.2.1 Power-controlled Energy-delay Adjustment

Here, we first derive the energy-efficient forwarding distance in the presence of a fast fading channel. Then, we propose a mechanism to decrease the end-to-end delay by increasing the transmit power.

Consider a node v_i forwarding a packet towards a destination actor a_k at distance D. We consider the link metric $E = 2E_{\text{elec}} + E_{\text{amp}} d^{\alpha}$, where α is the path loss propagation exponent ($2 \leq \alpha \leq 5$), E_{amp} is a constant $[J/(bits \cdot m^{\alpha})]$, and E_{elec} is the energy needed by the transceiver circuitry to transmit or receive one bit $[J/bits]$. The end-to-end energy consumption can then be expressed as

$$E_{\text{e-e}} = \sum_{(i,j) \in \mathcal{P}(v_i, a_k)} \left(\frac{P_{ij}^t}{r} + 2E_{\text{elec}} \right), \tag{3}$$

where $\mathcal{P}(v_i, a_k)$ represents the path between v_i and a_k. Ideally, the end-to-end energy consumption is minimized when data are forwarded on a set of nodes located on the line connecting the source and the destination, equally spaced with internode distance d^{opt}. By considering retransmissions, we obtain

$$E_{\text{e-e}}^{\min} = \min_{d, E_{\text{marg}}} \left\{ \frac{D}{d_{ij}} \cdot [2E_{\text{elec}} + (E_{\text{marg}} + E_{\text{amp}}) \cdot d_{ij}^{\alpha}] \cdot \mathcal{N}_{ij}^{R} \right\}. \tag{4}$$

The values for (d, E_{marg}) that minimize the above expression can be found by solving the nonlinear system $\nabla E_{\text{e-e}} = \mathbf{0}$, i.e., $[\frac{\partial E_{\text{e-e}}}{\partial d}, \frac{\partial E_{\text{e-e}}}{\partial E_{\text{marg}}}] = [0, 0]$, to find the stationary points of the function. Note that the *optimal forwarding distance* d^{opt} is independent of D, i.e., the distance between the forwarding node and the intended destination. The expression can be interpreted as the optimal trade-off between distance-independent and distance-dependent energy consumption, A practical forwarding rule should intuitively select the next hop with minimal distance from this point. However, it can be demonstrated that for values of α (path loss exponent) higher than 3.5, the expected energy consumption increases excessively when the next hop is closer than the optimal forwarding point to the destination. Hence, in this case, the next hop is selected as the closest node to the optimal forwarding point, among those that are not closer to the destination than the optimal forwarding point.

The reliability can be controlled by means of actor feedback messages. We adopt a conservative approach. When an event occurs, all sensors start transmitting with the maximum forwarding range. Then, according to the actor feedback on the observed reliability, sensors may decrease their forwarding range until either the reliability is close to the required event reliability threshold r_{th}, or until the optimal forwarding range is reached. Transmitting closer than the optimal forwarding range leads to high delay and high energy consumption, and is thus avoided. When the observed reliability is low even with the longest forwarding ranges, the actor initiates procedures for network layer congestion control, as explained in Section 3.2.2.

3.2.2 Network Layer Actor-driven Congestion Control

We propose to detect congestion at the actor receiving data and redirecting traffic to other, less congested nodes. Whenever an actor a_i detects very low reliability [36], caused by excessive delays and packet drops, it selects another actor to re-route the traffic from half of the sensors in its Voronoi cell to that actor. Each actor a_k is assigned by a_i a weight w_k, which measures its suitability to become a recipient for the traffic generated in the portion of the event area which a_i is receiving data from. The weight w_k, which is low for better-suited actors, is calculated as the weighted sum of the three factors, $w_k = \frac{c_\eta \eta_k + c_\delta \delta_k + c_\Delta \Delta_k}{c_\eta + c_\delta + c_\Delta}$, with weights c_η, c_δ, c_Δ. As a design choice, we set $c_\eta \geq c_\delta \geq c_\Delta$.

1) *Congestion factor* η_k, $0 \leq \eta_k \leq 1$. This normalized value reflects the reliability observed at actor a_k, i.e., $\eta_k = 1$ if $r < r_{th} - \epsilon$, it monotonically decreases as $r - r_{th}$ increases, and $\eta_k = 0$ for actors that are not receiving traffic. Here, ϵ represents a suitable margin on the reliability to avoid instability.

2) *Directivity factor* δ_k, that reflects the relative angular position of actor a_k with respect to actor a_i and the center of event area.

3) *Distance factor* Δ_k, which is the distance of the actor from the center of the event $C_{ev,i}$ normalized to the diameter of the monitored area, i.e., $\Delta_k = 1$ when the distance is maximal;

A congested actor a_i selects the optimal actor a_{k*} with minimum weight w_{k*}. Then, actor a_i calculates and advertises a new *virtual position* $\mathbf{x}_{\mathbf{k*}}^{\mathbf{virt}}$ for a_{k*} to the sensors in its Voronoi cell. The virtual position is forced to be on the line connecting the real position of the actor $\mathbf{x_{k*}}$ and the center of the event area $C_{ev,i}$, and

corresponds to the point such that half of the sensors in $C_{ev,i}$ are closer to a_i, while the other half is closer to a_{k*}. Each sensor will select its recipient actor, using for actor a_{k*} the virtual position $\mathbf{x}_{\mathbf{k*}}^{\mathbf{virt}}$, while the real position $\mathbf{x}_{\mathbf{k*}}$ is still used to perform the actual forwarding function. The concept of virtual position allows to optimally partition the sensors in such a way that only those that are closer to a_{k*} redirect their traffic to it, and provides a compact way to notify the sensors. The procedure is applied recursively by actors that are still congested after splitting the traffic in two.

3.3 Actor-Actor Coordination

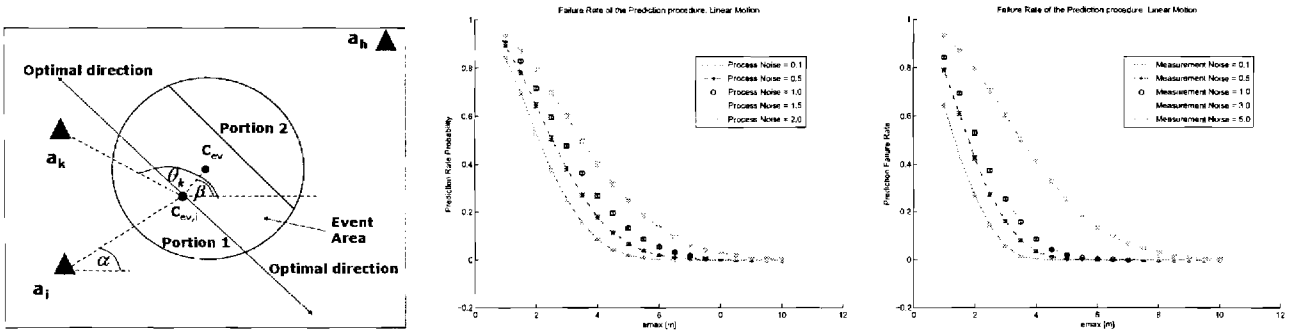
The objective of the multi-actor task allocation problem is to coordinate the mobility. In particular, it selects the best actor(s) to form the *actor team*, and to control their motion toward the action area. Our previous work [36] assumes that static actors are only able to act within a circular area defined by their action range. Hence, it is not suitable for WSANs with mobile actors. Moreover, in [36] reallocation of resources to face multiple events is not considered. Here, we introduce a more general framework and remove these assumptions.

According to the event features collected from the event area, each occurring event ω in the *event space* Ω can be characterized by the tuple $\mathcal{E}^{(\omega)} = \{F^{(\omega)}, Pr^{(\omega)}, A^{(\omega)}, S^{(\omega)}, I^{(\omega)}, D^{(\omega)}\}$, where $F^{(\omega)}$ describes the *event type*, i.e., the class the event belongs to, $Pr^{(\omega)}$ the *priority*, $A^{(\omega)}[m^2]$ the *event area*, $S^{(\omega)}[m^s]$ and $I^{(\omega)}[J/m^2]$ the *scope* (the action area) and *intensity*, respectively, and $D^{(\omega)}[s]$ the *action completion bound*, i.e., the maximum allowed time from the instant when the event is sensed to the instant when the associated action needs to be completed. These characteristics, which define each occurring event, are distributively reconstructed by the actors that receive sensor information, and constitute inputs to the multi-actor task allocation problem. In particular, the multi-actor allocation problem consists of selecting a *team of actors* and their *velocity* to optimally divide the action workload, so as to minimize the energy required to complete the action, while respecting the *action completion bound*. Although actors are resource-rich nodes, the order of magnitude of the energy required for actions and for movements is higher than that required for communication. Hence, it is important to save action and movement energy to extend the lifetime of actors. In [35], we formulate the multi-actor allocation problem as a *Mixed Integer Non-Linear Program* (MINLP).

3.4 Performance Results

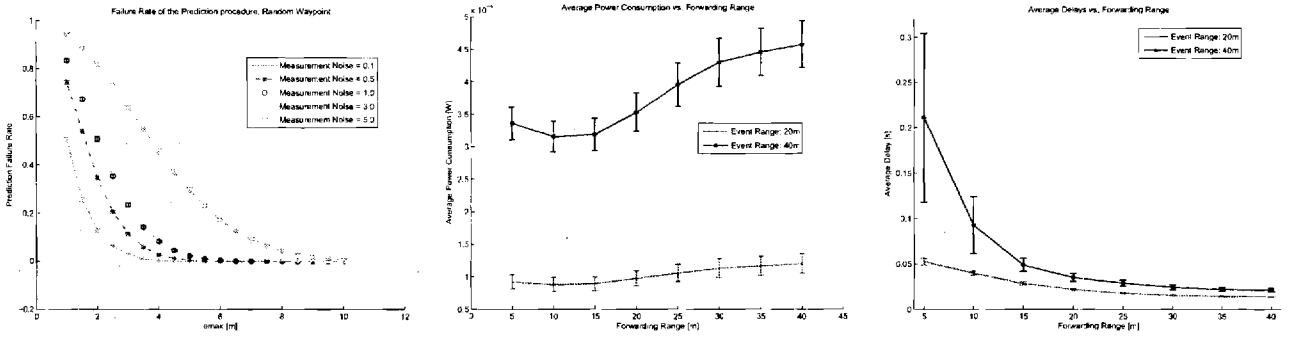
Performance results shown in this section are obtained with the sensor-actor simulator that we developed within the J-SIM framework [2]. First, we discuss results relevant to the prediction procedure described in Section 3.1.2. Actors move according to the model described in Section 3.1. In the first set of simulations, each actor selects a target destination and moves at constant speed to reach it. The actor implements a proportional controller that generates input commands to compensate for the process noise (random acceleration) by reestablishing the correct direction and speed. At each step, the actor measures its position (which is affected by measurement noise), filters the data, and decides whether an update needs to be sent.

In Fig. 7 (b-c) we report the *failure rate* of the prediction procedure, with varying values for e_{max}, and



(a) (b) (c)

Figure 7: Calculation of the directivity factor δ_i (a), Failure rate of the prediction procedure, with Linear Motion (b-c).



(a) (b) (c)

Figure 8: Failure rate of the prediction procedure with Random Waypoint Motion (a), average power consumption (b) and delay (c) vs. forwarding range.

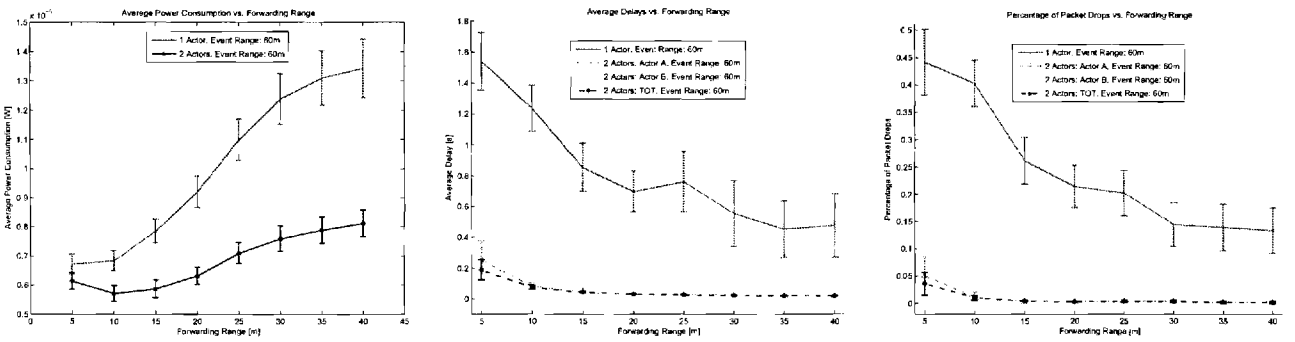
for different values of the process noise (random acceleration). The failure rate is defined as the number of location updates sent over all measurements taken at the actor. Each figure reports results averaged over different simulation scenarios, with 95% confidence intervals. In Fig. 7(b) we report the failure rate with varying process noise, while in Fig. 7(c) we show the failure rate with varying measurement noise. In the range of values analyzed, which corresponds to realistic motion scenarios, it is shown that if it is possible to accept a localization error of 5 m for the actors, which is reasonable being around 10% of the transmission range, the prediction at the sensors allows the actor to avoid 75% and more location updates, with proportional energy savings at the sensors. In the second set of simulations, reported in Fig. 8(a), actors select several different destinations during each simulation, similarly to a (perturbed) Random Waypoint model. The failure rate is only slightly higher, which shows that the prediction procedure proposed is effective even when complicated movement patterns are in place, and shows good robustness properties against noise.

As far as sensor-actor communication is concerned, sensors implement the geographical forwarding algorithm described in Section 3.2. The MAC layer is based on CSMA/CA. At the physical layer, we implemented

our power control procedure and set bandwidth and power consumption parameters similar to IEEE 802.15.4 compliant radios according to the Chipcon CC2420 datasheet. The monitored area is a 200 mx200 m square, with 200 randomly deployed sensors. The maximum transmission range of sensors is set to 40 m, and the bandwidth to 250 kbit/s. Sensors send 56 byte long packets with a reporting rate of 1 packet/s, and the size of the queues is set to 20 packets. We perform terminating simulations that last 400 s, average over different random topologies, and show 95% confidence intervals.

In Figs. 8 (b-c), we show a comparison of the average power consumption (b) and delay (c) with increasing forwarding range. Sensors inside the event area report measurements to the actor. The *event area* is circular and centered at $(100, 100)$ m. The figures report simulation runs for the cases of low and moderate traffic, i.e, the *event range* is equal to 20 m and 40 m around the center, respectively. In the first case, on average 7 sensors reside in the event area, while in the second case there are around 25 sources. In Figs. 8(b-c) we show that in situations of low and moderate traffic, which are common in sensor networks, the end-to-end delay can be consistently decreased by increasing the forwarding range. This is an important trade-off that has not been thoroughly explored so far. Clearly, this is paid with increased power consumption with respect to the optimal values.

Figure 9 refers to a high traffic scenario. The event range is set to 60 m, which corresponds to 57 sources on average. The event area lies completely in the Voronoi cell of a single actor. We compare energy consumption, delay, and packet drops when 1 or 2 actors receive the traffic generated in the event area, i.e., with or without the congestion control procedure devised in Section 3.2.2. We observe the following behavior. In the first case (no congestion control), the event area itself is congested, and a high percentage of packets are dropped (between 15% and 40%) (Fig. 9(c)), while the end-to-end delays increase to about 1 s and are not easily controlled by changing the forwarding range. Note that packets are dropped mostly in the event area due to multiple collisions at the MAC layer. Closer to the actor, the traffic is decreased due to earlier drops, and fewer nodes try to transmit simultaneously. Conversely, congestion can be dramatically decreased when the proposed congestion control procedure divides the event data between two actors. This is due to the fact that most of the congestion and packet drops occur in the event area, where many nodes try to transmit simultaneously, with the consequent drops due to simultaneous transmissions. This is dramatically improved when a second actor on the opposite side of the event area receives data, since traffic is diverted from the event area. The percentage of packets dropped is close to nil (see Fig. 9(b)), delays are two orders of magnitude lower and can be regulated with power control. Importantly, even though the second actor is farther (thus, in theory, suboptimal) from the event area, and although without congestion control packets are dropped early on their source-actor path, the power consumption is also decreased by the congestion control procedure, mostly due to reduced packet retransmissions at the MAC layer.



(a) (b) (c)

Figure 9: Energy consumption (a), delay (b), and packet drops (c) vs. forwarding range.

4 A Real-Time and Reliable Transport (RT)^2 Protocol for Wireless Sensor and Actor Networks

4.1 Overview

Wireless Sensor and Actor Networks (WSANs) are characterized by the collective effort of densely deployed sensor nodes and sparsely deployed actor nodes. In WSANs, sensor nodes collect information about the physical world, while actors take action decisions and perform appropriate actions upon the environment. The existing and potential applications of WSANs span a very wide range, including real-time target tracking, homeland security, battlefield surveillance, and biological or chemical attack detection [5]. Realization of these currently designed and envisioned applications, however, directly depends on real-time and reliable communication capabilities of the deployed sensor/actor network.

Recently, there has been considerable amount of research efforts, which have yielded many promising communication protocols for wireless sensor networks (WSNs) [4], [6], [49], [60], [9]. The common feature of these protocols is that they mainly address the energy-efficient and reliable data communication requirements of WSN. However, in addition to the energy-efficiency and communication reliability, many proposed WSAN applications have strict delay bounds and hence mandate timely transport of the event features from the sensor field to the actor nodes [5]. Consequently, the unique features and application requirements of WSANs call for a real-time and reliable data transport solution.

To address this need, in [20], we introduce a real-time and reliable transport (RT)^2 protocol for WSANs. (RT)^2 is a novel transport solution that seeks to *achieve reliable and timely event detection with minimum possible energy consumption and no congestion*. It enables the applications to perform right actions timely by exploiting both the correlation and the collaborative nature of WSANs. Furthermore, (RT)^2 addresses heterogenous reliability requirements of both sensor-actor and actor-actor communication. More specifically, for sensor-actor communication, unlike traditional end-to-end reliability notions, (RT)^2 defines *delay-constrained*

event reliability notion based on both *event-to-action delay bounds* and event reliability objectives. On the other hand, for actor-actor communication, it introduces 100% packet-level reliability mechanisms in order to avoid inaccurate action decisions in the deployment field. This way, the $(RT)^2$ protocol simultaneously addresses event transport reliability and timely action performance objectives of WSANs.

In general, compared to the existing transport layer proposals in the related literature, the main contribution of $(RT)^2$ is that *it concurrently provides real-time communication support and addresses heterogeneous transport reliability requirements* for typical WSAN applications involving reliable event detection and timely action objectives within a certain delay bound. To this end, the notion of *delay-constrained event reliability* distinguishes $(RT)^2$ from other existing transport solutions proposed for wireless ad hoc and sensor networks. To the best of our knowledge, reliable event transport has not been studied from this perspective before and hence $(RT)^2$ is the first solution attempt simultaneously addressing the real-time and reliable event transport and action performance objectives of WSANs.

In the following sections, we first describe the characteristics and challenges of WSANs and then based on these characteristics, we discuss the main design components of the $(RT)^2$ protocol in detail.

4.2 $(RT)^2$ Protocol Design Principles

In WSANs, the collaborative operation of the sensor nodes enables *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 the process of transmission of event features from the sensor nodes to the actor nodes as *sensor-actor communication*. Once an event has been detected in the deployment field, the actors need to communicate with each other to make a decision on the most appropriate way to collaboratively perform the action. We refer to this process as *actor-actor communication*. Therefore, the operation of the WSANs can be considered as a timely event detection, decision and acting loop. Next, we describe the details of the design principles and protocol operation of $(RT)^2$ for both sensor-actor and actor-actor communication.

4.2.1 Reliable Event Transport

The $(RT)^2$ protocol is equipped with different reliability functionalities in order to address heterogenous requirements of both sensor-actor and actor-actor communication. The main features of these reliability functionalities are described in the following.

4.2.2 Sensor-Actor Transport Reliability

In WSANs, sensor-actor transport does not require 100% reliability due to the correlation among the sensor readings [4],[58]. Hence, conventional end-to-end reliability definitions and solutions would only lead to over-

utilization of scarce sensor resources. On the other hand, the absence of reliable transport mechanism altogether can seriously impair event detection. Thus, the sensor-actor transport paradigm requires a collective *event transport reliability* notion rather than the traditional end-to-end reliability notions. The $(RT)^2$ protocol also considers the new notion of *event-to-action delay bound* (described in Section 4.3) to meet the application-specific deadlines. Based on both event transport reliability and event-to-action delay bound notions, we introduce the following definitions:

- The *observed delay-constrained event reliability* (DR_i) is the number of received data packets within a certain delay bound at the actor node in a decision interval i.
- The *desired delay-constrained event reliability* (DR^*) is the minimum number of data packets required for reliable event detection within a certain application-specific delay bound.
- The *delay-constrained reliability indicator* (δ_i) is the ratio of the observed and desired delay-constrained event reliabilities, i.e., $\delta_i = DR_i/DR^*$.

Based on the packets generated by the sensor nodes in the event area, the event features are estimated and DR_i is observed at each decision interval i to determine the necessary action. If the observed delay constrained event reliability is higher than the reliability bound, i.e., $DR_i > DR^*$, then the event is deemed to be reliably detected within a certain delay bound. Otherwise, appropriate action needs to be taken to assure the desired reliability level in sensor-actor communication. Therefore, sensor-actor transport reliability problem in WSANs is to *configure the reporting rate, f, of source nodes so as to achieve the required event detection reliability, DR^*, at the actor node within the application-specific delay bound.*

4.2.3 Actor-Actor Transport Reliability

In WSANs, a reliable and timely actor-actor ad hoc communication is also required to collaboratively perform the right action upon the sensed phenomena [5]. The $(RT)^2$ protocol simultaneously incorporates *adaptive rate-based transmission control and (SACK)-based reliability mechanism* to achieve 100% packet reliability in the required ad hoc communication. To achieve this objective, $(RT)^2$ protocol relies upon new feedback based congestion control mechanisms and probe packets to recover from subsequent losses and selective-acknowledgments (SACK) to detect any holes in the received data stream. These algorithms are shown to be beneficial and effective in recovering from multiple packet losses in one round-trip time (RTT) [50]. Furthermore, to prevent congestion in the reverse path, SACK packets are delayed in the receiver, i.e., one SACK packet for every d data packets received. Hence, this delayed SACK strategy of $(RT)^2$ protocol enables the receiver to control the amount of the reverse path traffic accordingly. Next, event-to-action delay bound notion of the $(RT)^2$ protocol is explained in detail.

4.3 Real-Time Event Transport

To assure accurate and timely action on the sensed phenomena, it is imperative that the event is sensed, transported to the actor node and the required action is performed within a certain delay bound. We call this *event-to-action delay bound*, Δ_{e2a}, which is specific to application requirements and must be met so that the overall objective of the sensor/actor network is achieved. The event-to-action delay bound Δ_{e2a}, has three main components as outlined below:

1. **Event transport delay (Γ^{tran}):** It is mainly defined as the time between when the event occurs and when it is reliably transported to the actor node. Therefore, it involves the following delay components:

   (a) *Buffering delay ($t_{b,i}$):* It is the time spent by a data packet in the routing queue of an intermediate forwarding sensor node i. It depends on the current network load and transmission rate of each sensor node.

   (b) *Channel access delay ($t_{c,i}$):* It is the time spent by the sensor node i to capture the channel for transmission of the data packet generated by the detection of the event. It depends on the channel access scheme in use, node density and the current network load.

   (c) *Transmission delay ($t_{t,i}$):* It is the time spent by the sensor node i to transmit the data packet over the wireless channel. It can be calculated using transmission rate and the length of the data packet.

   (d) *Propagation delay ($t_{p,i}$):* It is the propagation latency of the data packet to reach the next hop over the wireless channel. It mainly depends on the distance and channel conditions between the sender and receiver.

2. **Event processing delay (Γ^{proc}):** This is the processing delay experienced at the actor node when the desired features of event are estimated using the data packets received from the sensor field. This may include a certain decision interval [4] during which the actor node waits to receive adequate samples from the sensor nodes.

3. **Action delay (Γ^{act}):** The action delay is the time it takes from the instant that event is reliably detected at the actor node to the instant that the actual action is taken. It is composed of the *task assignment delay*, i.e., time to select the best set of actors for the task and the *action execution delay*, i.e., time to actually perform the action.

More specifically, while event transport delay Γ^{tran} and event processing delay Γ^{proc} occur during sensor-actor communication, action delay Γ^{act} is resulted from actor-actor communication in the deployment field. Let Δ_{e2a} be the event-to-action delay bound for the data packet generated by the detection of event. Then, for a timely action, it is necessary that

$$\Delta_{e2a} \geq \Gamma^{tran} + \Gamma^{proc} + \Gamma^{act} \tag{5}$$

is satisfied. Here, Γ^{tran} is clearly a function of $t_{b,i}$, $t_{c,i}$, $t_{t,i}$, $t_{p,i}$, and $\hat{N}$, where $\hat{N}$ is the average hop count from the source nodes to the actor node.

Note that Γ^{tran} is directly affected by the current load and the congestion level in the network. In addition, the network load depends on the event reporting frequency, f, which is used by the sensor nodes to send their readings of the event. Hence, the main delay component that depends on the congestion control and thus, can be controlled to a certain extent is the event transport delay, i.e., Γ^{tran}. More specifically, the buffering delay, i.e., $t_{b,i}$, directly depends on the transport rate of the event, queue management and service discipline employed at each sensor node i.

In addition, for the events occurring at further distances to the actor node, the average number of hops that event data packets traverse, $\hat{N}$, increases. Thus, it is more difficult to provide event-to-action delay bound for further event packets compared to closer ones. Considering that the per-hop propagation delay, $t_{p,i}$, does not vary[4], then the buffering delay, $t_{b,i}$, must be controlled, i.e., decreased, in order to compensate the increase in the event transport delay so that the event-to-action delay bound is met.

To accomplish this objective, the (RT)^2 protocol introduces *Time Critical Event First* (TCEF) scheduling policy. In fact, TCEF policy applies the general principles of Earliest Deadline First service discipline on each sensor node, which is shown to be the optimal scheduling policy when real-time deadlines of the system are considered [16],[44]. However, we also integrate some novel mechanisms so as to fit it to unique challenges of sensor networks [20]. Note that although TCEF policy makes it possible to meet deadlines in the normal operating conditions of the network, in case of severe network congestion, it may become insufficient to provide delay-constrained event reliability. Hence, in addition to TCEF scheduling, (RT)^2 considers the event-to-sink delay bounds and congestion conditions in its reporting rate update policies to assure timely and reliable event transport.

4.4 Congestion Detection and Control

In WSANs, because of the memory limitations of the sensor nodes and limited capacity of shared wireless medium, congestion might be experienced in the network. Congestion leads to both waste of communication and energy resources of the sensor nodes and also hampers the event detection reliability because of packet losses [4]. Hence, it is mandatory to address the congestion in the sensor field to achieve real-time and reliable event detection and minimize energy consumption. However, the conventional sender-based congestion detection methods for end-to-end congestion control purposes cannot be applied here. The reason lies in the notion of delay-constrained event reliability rather than end-to-end reliability. Only the actor node, and not any of the sensor nodes, can determine the delay-constrained reliability indicator $\delta_i = DR_i/DR^*$, and act accordingly.

In addition, for efficient congestion detection in WSANs, the sensor nodes should be aware of the network channel condition around them, since the communication medium is shared and might be congested with the network traffic among other sensor nodes in the neighborhood [26]. Therefore, because of shared communica-

[4]While the channel access delay can also be controlled to a certain extent via priority-based QoS-aware MAC protocols [6], we do not assume the presence of such MAC protocol.

Table 1: Network Operation Regions Based on Congestion and Delay-Constrained Event Reliability

Decision Boundaries	Characteristic Region
$\delta < 1-\beta$ and CN=1	Low Reliability and Congestion
$\delta > 1+\beta$ and CN=1	Early Reliability and Congestion
$\delta < 1-\beta$ and CN=0	Low Reliability and No Congestion
$\delta > 1+\beta$ and CN=0	Early Reliability and No Congestion
$1-\beta < \delta < 1+\beta$ and CN=0	Optimal Operating Region

tion medium nature of WSANs, the sensor nodes can experience congestion even if their buffer occupancy is small.

In this regard, $(RT)^2$ uses a *combined congestion detection* mechanism based on both average node delay calculation and local buffer level monitoring of the sensor nodes to accurately detect congestion in the network. Note that average node delay at the sensor node gives an idea about the contention around the sensor node, i.e., how busy the surrounding vicinity of the sensor node. In combined congestion detection mechanism of $(RT)^2$ protocol, any sensor node whose buffer overflows due to excessive incoming packets or average node delay is above a certain delay threshold value is said to be congested and it informs the congestion situation to the actor node.[5] More specifically, the actor node is notified by the upcoming congestion condition in the network by utilizing the *Congestion Notification* (CN) bit in the header of the event packet transmitted from sensors to the actor node. Therefore, if the actor node receives event packets whose CN bit is marked, it infers that congestion is experienced in the last decision interval. In conjunction with the delay-constrained reliability indicator, δ_i, the actor node can determine the current network condition and dynamically adjust the reporting frequency of the sensor nodes.

4.5 $(RT)^2$ Protocol Operation

Unlike traditional networks, the sensor/actor network paradigm necessitates that the event features are collaboratively estimated within a certain reliability and real-time delay bound. To achieve this objective with maximum resource efficiency, the $(RT)^2$ protocol addresses heterogeneous communication requirements of both sensor-actor and actor-actor communications. Recall that while sensor-actor communication may not require 100% reliability due to the correlation among the sensor readings [4],[58], actor-actor communication requires 100% reliability to make a decision on the most appropriate way to perform the action.

More specifically, during sensor-actor communication, the objective of $(RT)^2$ is to determine the reporting frequency, f, to maintain the desired event estimation with minimum energy expenditure. To accomplish this objective, the actor nodes update the reporting frequency of the sensors in conjunction with the congestion notification information (CN bit) and delay-constrained reliability indicator, δ. This updating process is repeated until the optimal operating point is found, i.e., adequate reliability and no congestion condition is obtained. For

[5]To avoid reacting to transient network behavior and to increase the accuracy of congestion detection, the $(RT)^2$ protocol detects congestion, if the node delay measurements exceed a delay threshold more than a certain number of successive times.

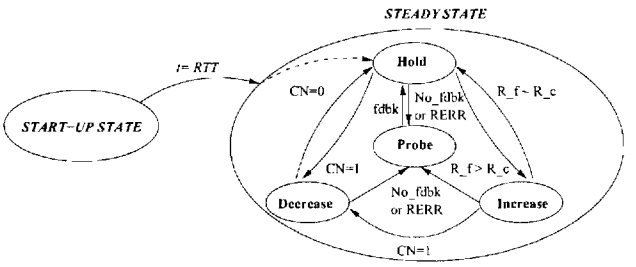


Figure 10: $(RT)^2$ state transition diagram for actor-actor communication.

practical purposes, we also define a tolerance level, β, for optimal operating point. Based on delay-constrained event reliability requirement and dynamic network conditions, $(RT)^2$ determines the characteristic regions, which identify the state of the network, as shown in Table I. According to these characteristic regions, the $(RT)^2$ protocol updates the reporting frequency, f, so that the number of sensor samples received in a decision interval i, i.e., DR_i, is adequate for reliable event-to-actor transport with minimum energy expenditure [20].

Note that for sensor-actor communication $(RT)^2$ exploits both the correlation and the collaborative nature of WSANs to adjust the reporting frequency of the sensors accordingly. To address different reliability requirements of actor-actor communication, $(RT)^2$ also incorporates adaptive rate-based transmission control and (SACK)-based reliability mechanism during actor-actor communication. More specifically, it periodically adjusts transmission rate based on bottleneck node information, i.e., congestion notification (CN), rate feedback (R_f) and route error ($RERR$). Here, $(RT)^2$ benefits from both cross-layer interactions and intermediate node feedback information.

In Fig. 10, the $(RT)^2$ protocol state diagram for actor-actor communication is shown. As shown in Fig. 10, the protocol operation is composed of two main states: i) start-up state, ii) steady state. Specifically, the steady state is composed of four sub-states, i.e., increase, decrease, maintain and probe. Basically, $(RT)^2$ relies on feedback from the intermediate network nodes. Also, its increase is more aggressive than that of TCP, decrease is less conservative than that of TCP, and more importantly operates in a maintain state when network conditions do not change considerably. In addition, to meet the application-specific delay bounds, $(RT)^2$ determines the minimum transmission rate (R_{min}) according to the remaining time to event-to-action delay bound. This way, the data rate is dynamically adjusted based on both the current conditions of the data path and event-to-action delay bounds. The details of the $(RT)^2$ protocol operation during sensor-actor and actor-actor communications can be found in [20].

4.6 $(RT)^2$ Performance Evaluation

Here, we present the performance evaluation of the $(RT)^2$ protocol. In Section 4.6.1, we report the performance results for the sensor-actor communication, while in Section 4.6.2, we discuss the performance results for the actor-actor communication. The details of $(RT)^2$ performance results for various network parameters can be found in [20].

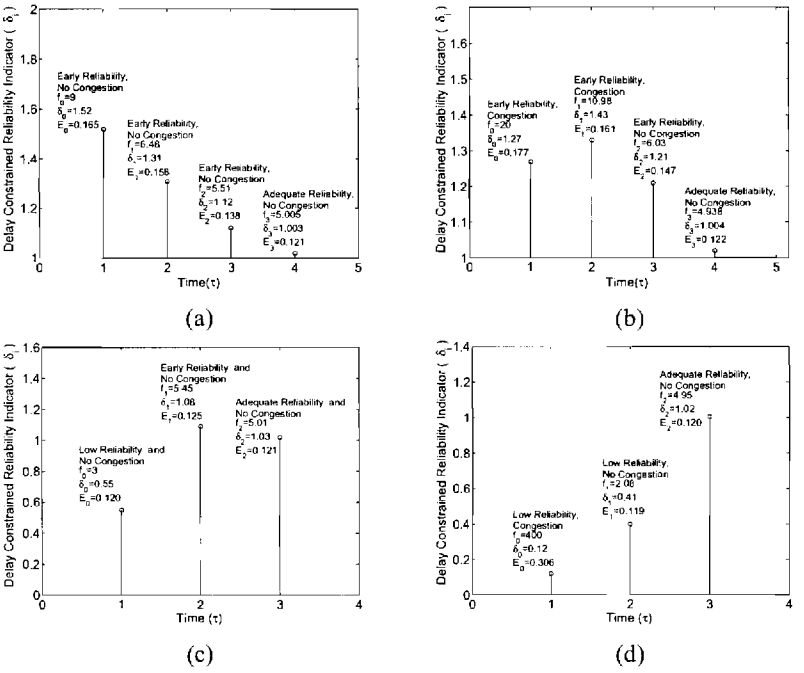


Figure 11: The $(RT)^2$ protocol trace, when (a) early reliability and no congestion, (b) early reliability and congestion, (c) low reliability and no congestion, (d) low reliability and congestion, is observed.

4.6.1 Performance Results for Sensor-Actor Communication

To evaluate the performance of the $(RT)^2$ protocol during sensor-actor communication, we developed an evaluation environment using ns-2 [52]. For sensor-actor communication scenario, 200 sensor nodes were randomly positioned in a 200m x 200m sensor field. Also, event centers were randomly chosen and all sensor nodes within the event radius behave as sources for that event. We run 10 experiments for each simulation configuration. Each data point on the graphs is averaged over 10 simulation runs. The details of sensor node and simulation configurations can be found in [20].

The $(RT)^2$ protocol convergence results are shown in Fig. 11 for different initial network conditions. As observed in Fig. 11, $(RT)^2$ protocol converges to (Adequate reliability, No congestion) condition starting from any of the other initial network conditions discussed in Section 4.5. Thus, $(RT)^2$ is self-configuring and can perform efficiently under random, dynamic topology frequently encountered in WSAN applications. Moreover, the average energy consumed per packet during sensor-actor communication, i.e., (E_i), is also observed. As shown in Fig. 11, E_i decreases as the (no congestion, adequate reliability) state is approached which shows that energy consumption of the sensor nodes is also decreased while providing reliability constraints and delay bounds. Due to energy limitations of sensors, this result is also important for the proper operation of WSAN. Performance of reporting frequency update policies for sensor-actor communication are given as the trace values and states listed within Fig. 11.

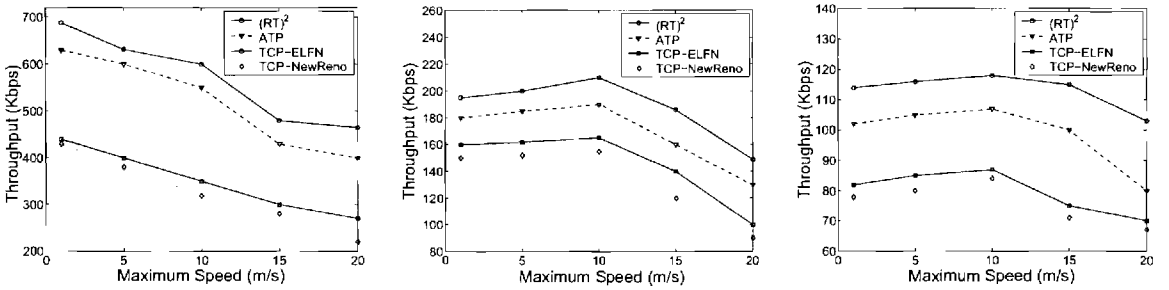


Figure 12: Aggregate throughput for (a) 1 flow connection, (b) 5 flow connection, (c) 10 flow connection, when the maximum speed of the actors are varying.

4.6.2 Performance Results for Actor-Actor Communication

In this section, we present the performance results of the $(RT)^2$ protocol during actor-actor communication. For the simulations, we set up an evaluation environment using ns-2 [52]. The simulations for this scenario are performed for a 200m x 200m field with 10 actor nodes, distributed randomly over the field. In addition, to take into account the mobility of the actors during actor-actor communication, we have used the random way-point model. In this mobility model, we consider maximum speeds of $1m/s$, $5m/s$, $10m/s$, $15m/s$ and $20m/s$ for mobile actor nodes.

For actor-actor communication scenario, the performance of the $(RT)^2$ protocol is evaluated and compared against TCP-NewReno, TCP-ELFN [25] and ATP [50]. In Fig. 12, we present the aggregate throughput results of the $(RT)^2$ protocol and other ad hoc transport protocols under comparison. Here, different number of flow connections are used and source-destination pairs are randomly chosen from 10 actor nodes. In terms of aggregate throughput, the $(RT)^2$ protocol outperforms other transport protocols under comparison, since $(RT)^2$ dynamically shapes data traffic according to the channel condition and intermediate node feedbacks. In addition, proper reaction of $(RT)^2$ to congestion and non-congestion related losses, such as route failures, avoids any performance degradation during actor-actor communication. For example, for 5 flow connection and $10m/s$ speed, we obtain that the aggregate throughput achieved by $(RT)^2$ during actor-actor communication is around 40%, 30% and 15% higher than that of TCP-NewReno, TCP-ELFN and ATP, respectively. Note also that rate-based transport protocols, i.e., $(RT)^2$ and ATP, outperform window-based transport protocols, i.e., TCP-ELFN and TCP-NewReno, mainly because rate-based schemes capture the available bandwidth more quickly compared to window-based schemes. We have evaluated the performance of $(RT)^2$ in terms of various network parameters, such as end-to-end delay and data transfer time. We found that $(RT)^2$ outperforms other protocols under comparison. The details of these experiments can be found in [20].

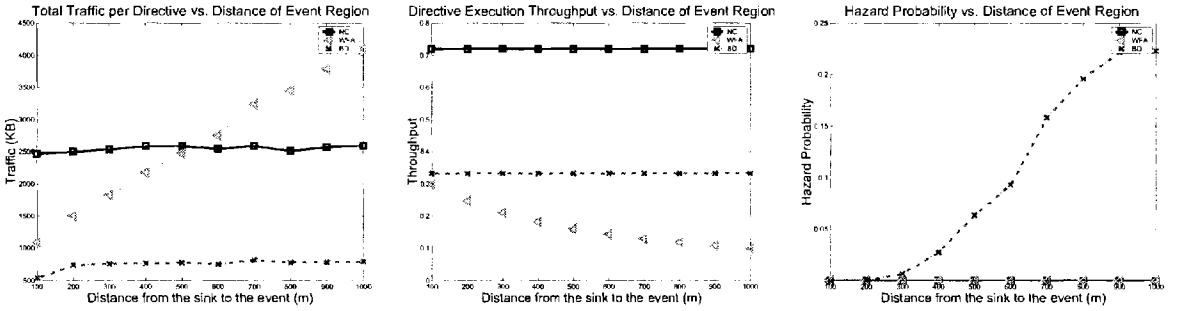
5 Hazard Avoidance in Wireless Sensor and Actor Networks

5.1 Problem

As mentioned above, the presence of read and write operations in WSANs leads to unique challenges that need to be addressed. Consider a simple example of a WSAN that uses fire-detector sensors along with water-sprinkler actors. Assume that the sink has issued a *command* directive[6] C to the actors to sprinkle water in response to sensor feedback about a fire. Now, after a certain period of time t, consider the sink to issue a *query* directive Q to check if the fire has been extinguished. If, for a certain region in the WSAN, Q is delivered and executed *before* C by the network, the corresponding response by the sensors - that the fire still exists - will trigger an unnecessary reaction by the sink in the form of more directives to the actors to sprinkle more water. We refer to such problems where the execution order of directives is different from what the sink intends or expects it to be as *hazards*. Essentially, the inherent dependency between the actions performed by the sensors, and those performed by the actors, imposes a need for correctness of operations. Thus, a *hazard* is the *out-of-order execution of directives due to a lack of coordination between sensors, actors and the sink* that can potentially lead to undesirable changes in the physical environment. We first identify the different types of hazards in a WSAN environment. We then present the design of a distributed and localized approach called the Neighborhood Clock approach that addresses the hazards.

5.2 Types

We define three types of hazards and describe them with an illustration.



(a) Total Traffic per Directive (b) Directives Execution Throughput (c) Correctness

Figure 13: Performance under different sink-to-event distance

5.2.1 CAC Hazard

Definition: Consider a set of n directives, I_1,I_2,...,I_n. Let I_k and I_{k+1} be two dependent, sequential commands (i.e., I_{k+1} has to be executed strictly after I_k is executed) sent to two actors in the event region, A_x and A_y.

[6]We use the term *directives* to generically refer to both commands and queries.

Let $E_{k,x}$ denote the execution of the command I_k by actor A_x finishing at time, T_1, and $E_{k+1,y}$ denote the execution of command I_{k+1} by actor A_y starting at time T_2. A command-after-command (CAC) hazard occurs when: $T_2 < T_1$.

Illustration (see figure 14): Consider two dependent, sequential commands, $D1$ and $D2$, issued to two different actors in the event region. Consider the case when $D2$, through path $C1$, arrives before the directive $D1$ through path $C2$. This results in a CAC hazard that may lead to undesirable changes in the environment.

5.2.2 QAC Hazard

Definition: Let I_k and I_{k+1} be two related, sequential directives, with I_k being a command sent to an event region containing an actor, A_x and I_{k+1} being a query sent to the event region comprising a sensor, S_y. Let $E_{k,x}$ denote the execution of the command I_k by the actor S_x completing at time, T_1 and $R_{k+1,y}$ denote the response to query I_{k+1} by sensor S_y initiated at time, T_2. A query-after-command (QAC) hazard occurs when: $T_2 < T_1$.

Illustration (see figure 14): Let $D1$ and $D2$ be sequential directives as described before. Consider the case when the $D2$, a query, reaches the sensors via path $Q2$ before directive 1 (received via path $C1$) was executed. This results in a QAC hazard, where a query issued after a command gets executed prior to the execution of the command.

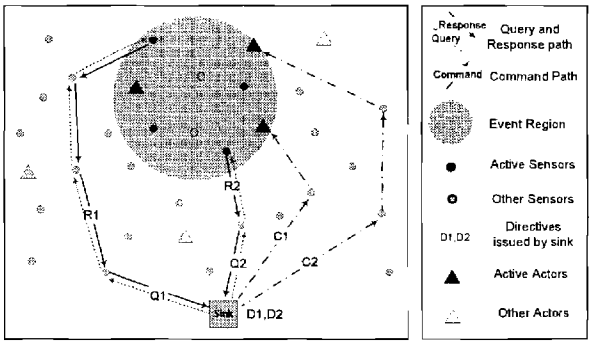


Figure 14: Hazard Illustration

5.2.3 CAQ Hazard

Definition: Let I_k and I_{k+1} be two related, sequential directives, with I_k being a query sent to the event region consisting of sensor, S_x, and I_{k+1} being a command sent to the event region consisting of actor, A_y. Let $R_{k,x}$ denote the response to query I_k by sensor S_x initiated at time, T_1 and $E_{k+1,y}$ denote the execution of command I_{k+1} by actor S_y starting at time, T_2. A command-after-query (CAQ) hazard occurs when: $T_2 < T_1$.

Illustration (see figure 14): Suppose a query ($D1$) was issued following which two responses arrive at the

sink at different times. The first response comes through path $R2$, based on which the sink issues $D2$. This command reaches the actor with a short delay via path $C1$. After that, the other response for the $D1$ comes through $R1$. In this case, the sink will not be able to determine whether the second response was initiated before or after $D2$ has been executed, resulting in a CAQ hazard.

5.3 Related Work

5.3.1 Pipelining

The problem considered in this section shares some similarities to pipelining of instructions in the computer architecture domain [24]. In order to resolve any dependencies within the instruction set several techniques have been proposed including instruction re-ordering and register re-allocation. This is philosophically similar to what we have tried to achieve in our approach in terms of increasing the parallelism in issuing directives. However, in WSANs, we not only have to maximize the directive level parallelism but also region-level parallelism.

5.3.2 Parallel Programming

The hazards in WSANs has some resemblance to the synchronization problem in the context of multiprogramming in the operating systems domain [47]. In parallel programming, software primitives such as semaphores and monitors are used to bring about synchronization. However, these approaches are not suitable in the context of WSANs.

5.3.3 Distributed Systems

The NC approach shares some ideas from the distributed systems area. A distributed system consists of a set of processes that cooperate to achieve a common goal, but do not share a common global memory. To capture the causality relationship between events, logical clock model is used [11]. Unlike in distributed systems, where the goal is to achieve global ordering for a set of processes, NC addresses the hazard problems only within the dependency region of an entity.

5.4 Approach

For hazard free operation, the following two observations are necessary and sufficient:

- *Any pair of dependent directives issued to entities*[7] *that do not have any overlapping execution regions*[8]

[7] We refer to sensors and actors with the common term entity.

[8] We refer to sensing region of a sensor and acting region of an actor with the generic term, execution region.

can be executed concurrently across the two entities, although the relative ordering must be preserved within each entity.

- *Any pair of dependent directives issued to entities with overlapping execution regions needs to be ordered in the union of the two regions.*

Now, for a given entity, D_x, applying these rules pairwise with any other entity in the event region, we can define a region in the neighborhood of D_x called the *dependency region*, where perfect ordering is necessary. The dependency region of a sensor is the region with radius equal to the sum of sensing and acting range, while that of an actor is the region with radius as twice the acting range.

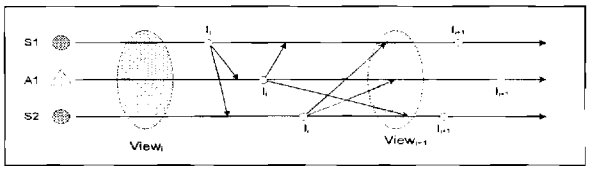


Figure 15: View Movement

In order to ensure hazard free operations, we introduce the Neighborhood clock (NC) approach that uses the notion of a *virtual clock* on every sensor and actor for ordering the directives within every *dependency* region. When the sink learns about an event, the sink creates a *reference clock* for that event, and initializes this clock to a unique start value, NC_0. All sensors and actors in the event region initialize their *neighborhood clock* by the initial reference clock value. Each entity, D_x, maintains its own *view* of the progress in the network, based on its neighborhood clock identifier, $NC(x)$, where the view number is set to be $NC(x) + 1$. NC functions by synchronizing the NC values of all neighborhood clocks. Each sensor and actor will move to the next view only after all other sensors and actors have moved into its current view. The progress of views within a dependency region for three entities is shown in Figure 15. Any entity, D_x, can execute a directive only if the NC value piggybacked is the same as the current view. Once an entity executes a directive, it sends a notification to other entities in the dependency region.

5.5 Results

We evaluated the NC approach with two baseline strategies: (i)Wait-For-All (WFA), and (ii) Bounded Delay (BD). In Wait For All (WFA) strategy, the sink issues the next directive only after it receives all the responses or notifications for the previous directive. For instance, if the sink sent out a command, it will wait for acknowledgements from all the actors before it issues another query or command. In BD, after issuing a query, the sink waits for time T_{W_s} before issuing the next directive. Similarly, after a command is sent, the sink waits for at least T_{W_a} prior to sending another directive.

Figure 13 shows the performance results of the three approaches for varying sink-to-event distances. We can see that in WFA has a much higher overhead in dealing with far-away events due to the fact that all the sensors and actors in the event region are required to respond back to the sink. As shown in Figure 13(a),

both BD and NC have (almost) constant traffic, which only increases slightly with increasing sink-to-event distance. This is because the average traffic in delivering the directive is almost a constant. Additionally in NC, the traffic generated within the dependency region will always be a constant. Figure 13(b) shows that NC has largest throughput. The throughput of WFA drops because the waiting time for issuing a directive increases with increasing sink-to-event distance. Unlike WFA, the throughput of NC and BD do not change with the sink-to-event distance, since the latency between the execution of successive directives does not depends on the distance of the event from the sink. Similar to that of increasing event region size, the hazard probability of BD is higher for a farther away event, which is shown in Figure 13(c). We have also evaluated the NC approach for a variety of other network conditions, and found that it out-performed the WFA and BD in terms of the directives execution throughput and provided 100% hazard-free operation.

6 Mutual Exclusion in Wireless Sensor and Actor Networks

6.1 Problem

In this part of the work, we identify the problem of "*mutual exclusion*", which is the need to act only once for any particular location and command. Consider a simple example of a WSAN that uses fire-detector sensors along with water-sprinkler actors. Assume that the sink has issued a *command* directive[9] C to the actors to sprinkle water in response to sensor feedback about a fire. Now, if two actors in the vicinity of each other which have overlapping acting regions, receive the same command, in the overlapped region, the action will be performed twice. While the undesired outcome in the above example is merely the wastage of water, depending upon the nature of the application, such outcomes can even be catastrophic (e.g. poison gas actors where one dose of the gas merely invalidates subject, but two doses can kill). We refer to this problem of providing mutually exclusive acting regions to cover an event region as the mutual exclusion problem and identify three different types of the problem in Section 6.2. In this context, we discuss the centralized approach needed to address this problem efficiently. We propose a distributed and fully localized *mutual exclusion* approach that addresses the problem for all three types.

6.2 Types

The conventional distributed mutual exclusion provides access to a shared critical resource among a group of processes [11]. It involves a group of processes, each of which intermittently requires access to a shared resource or a piece of code called the *critical section* (CS), where at most one process may be in the CS at any given time. However, the mutual exclusion problem in the case of WSANs is quite different and unique. In WSANs, mutual exclusion is defined with respect to both a directive and location, where given a command, I_i, and location, X_i, the execution of command I_i should happen in X_i exactly once by any one of the actors

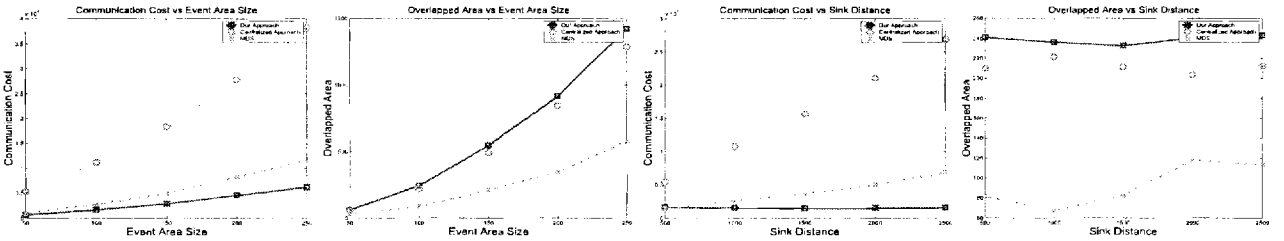
[9] We use the term *directives* to generically refer to both commands and queries.

(processes) in the vicinity of the region. Thus, the mutual exclusion problem for WSANs can be defined as follows: Given a set of actors, $a_1 \ldots a_i \ldots a_k$ in the event region, R, the mutual exclusion problem is to determine minimum subset of actors, S_a, called the *actor cover set* required to cover the entire event region such that there is minimal overlap, where overlap is defined by a benefit function, and each region is processed exactly once.

6.2.1 Conservation of Actor Resources

In this definition, it is necessary to maximize the non-overlapped acting regions of each actor within the event region in order to utilize the actor resources to the least extent. These resources could either correspond to the power level consumed by the actors or amount of resource utilized to perform actions. This definition is similar to the sensor coverage problem defined in [22].

Consider a fire extinguisher example, where the sensors are heat sensors that detect the presence of fire and sprinklers serve as actors to quench the fire. Consider the case where the amount of water available in the sprinklers is limited. In such a case, while dousing a fire, it is absolutely necessary to ensure that the there is no wastage of water while dousing a fire. In such a case, it is desirable that each actor selected in the event region, has non overlapping acting regions so that the number of actors selected to cover the event region is minimum. In other words, the new area covered by any actor that is added to the already existing cover set should maximize the non overlapped region of that actor.



(a) Total Traffic (b) Overlapped Area (c) Total Traffic (d) Overlapped Area

Figure 16: Performance under Different Event Size and Different Event Distances

6.2.2 Binary Decision Making

Here, it is necessary to maximize the non-overlap area in the actor regions while also reducing the amount of new overlap in the acting regions with already existing cover set. This scenario is desirable when applications take a binary stand with regards to the effect of excessive action in a region, where the action taken is either desirable level or excessive. In other words, there is a threshold for the desired level of action and any amount of action beyond this threshold is perceived as undesirable irrespective of the exact value above the threshold.

Consider an intruder detection and automated tranquilizer application, where the sensors are image sensors that detect the presence of an intruder, and the actors are poison gas actors where one dose of the gas merely

invalidates subject, but two doses can kill. In such a case, when the amount of poisonous gas injected in any area is twice the normal dosage or more, the effect of the action is the same and results in the death of the subject. In such an event, it is imperative to minimize any overlap the first time around as it will result in actions being taken twice. However, once an overlap is deemed as inevitable, it does not matter if there is any additional overlap in the same (overlapped) region by another actor in the cover set. In other words, it is desirable to let any new actor that is added to the existing cover set to overlap with already overlapped regions as long as it does minimizes the new overlap region.

6.2.3 Fine-Grained Decision Making

In this case, it is not only necessary to maximize the amount of non-overlapped region covered by each actor but also equally important to minimize the amount of overlapping regions (both old and new). This corresponds to the scenario where every overlap is deemed as excessive action. In other words, any action that takes place beyond the desired level is unacceptable and the net overlap should be minimized irrespective of whether it occurred in the same region or elsewhere.

Consider a fire extinguisher example mentioned above with heat sensors for sensing and sprinklers as actors. Let the appropriate level of action be described by one actor (sprinkler) acting on a fire event in any particular. Consider the case, when the acting ranges of sprinklers overlap. In such a case, the regions where the overlap occur will result in flood. If the overlap occurs multiple times, the flooding will be severe in those regions. In this scenario, apart from maximizing the non-overlapped region, it is absolutely necessary that the sum of all overlapped region is minimized irrespective of whether they are localized or otherwise.

6.3 Related Work

6.3.1 Sensor and Actor Networks

In [22], the authors propose centralized and distributed solutions for determining the minimum connected sensor cover in order to reduce the overall energy consumption of a pure wireless sensor network (WSN). This problem is different to mutual exclusion problem in WSANs, where there is no need for the actors to be connected. Also, there different types of mutual exclusion identified in the context of WSANs, are not required in a pure WSN environment. Further [22], does not try to minimize delay and hence does not incorporate any delay constraints in the design of their approaches. Also, [22] does not address any of the challenges that are unique to a WSAN environment. The authors in [36] have considered a different problem in WSANs pertaining to actor-actor coordination, where the goal is to determine the set of actors to cover an event region when the actors have different power levels. Here, the actor set is optimized to reduce the overall power consumption, which is different from the problem of mutual exclusion.

6.3.2 Mutual Exclusion in Ad hoc Networks

The distributed mutual exclusion problem has been identified in the context of ad hoc networks in the context of assignment of channels and shared resources [11, 31, 59]. However, these works do *not* conform to the definition of mutual exclusion in the context of WSANs.

6.4 Approach

In this section, we outline a greedy centralized approach for providing mutual exclusion that is $O(log(\Omega))$ factor to the optimal solution, where Ω is the number of actors with overlapping acting range with the event region. The approach is similar to the centralized approach for connected sensor cover presented in [22] with a few modifications. However, as opposed to that work, in this case it is not required that the actors be connected. Also, the selection criteria for every actor is defined based on the benefit function (VM) of the actor. We also mention the key design elements in order to extend the solution to a distributed approach called the *Neighborhood Backoff* (NB) Approach.

To describe briefly, the greedy algorithm works by selecting, at each stage, the actor with the maximum benefit function. The selected actor is added to the already selected actors at that stage. The benefit function is updated for all the actors that have overlapping acting regions based on the new values of the region covered, new non-overlapped region, new overlap region and old overlap region. The algorithm terminates when the selected set of actors cover the complete event region.

For a given sensor or actor, D_x, the maximum region within which another entity can have an impact on its sensing or acting range is defined to be the *dependency region*. The dependency region of a sensor can be defined as the region with radius equal to the sum of sensing and acting range ($Sensing\ Range + Acting\ Range$), while that of an actor is the region with radius as twice the acting range ($2 \cdot ActingRange$). The distributed approach called the NB approach is based on the following three key ideas:

- Determination of initial benefit function for each actor based on the directives issued by the sensors to the actors in its dependency region
- Emulation of the greedy centralized strategy at each actor by waiting for appropriate amount of time to execute a directive proportional to benefit function of that node. If the benefit function of a particular actor is large (close to its execution range), the waiting time will be relatively small and when the benefit function is very small, the waiting time will be relatively large.
- Updating the benefit function (and hence the waiting time for execution) for an actor when any actor within its dependency region has acted on a specific directive.

In essence, the NB approach is a randomized, distributed approach that approximates the greedy, centralized approach by adjusting the waiting time for acting based on the benefit function for the mutual exclusion problem, and adjusting the benefit function and waiting time when an actor in the dependency region has acted.

6.5 Results

This section evaluates the performances of our proposed approach (NB) with two basic strategies: Centralized Set Cover (CSC), the greedy centralized approach described in Section 6.4, and Minimum Dominating Set (MDS) [38].

6.5.1 Varying the Event Area Size

Figures 16 (a) and (b) shows the performance results of the three approaches under varying event area size. Our approach achieves the best performance in terms of overhead. As shown in Figure 16(a), with increasing event area size, the traffic per event of all three approaches increases. For CSC and MDS approaches, this is mainly because of the increase in the number of sensors and actors in the event area, which means more reports and commands are sent to or originated from the sink. For our approach, since each node has to receive notifications from all other nodes within its dependency region, the overhead is increasing. Figures 16(b) shows the result of overlapped action area. As we observe, our approach has only slightly worse performance than centralized one but can achieve 100% correctness.

6.5.2 Varying the Distance from the Sink to the Event Center

Figure 16 (c) and (d) shows the performance results of the three approaches for varying sink-to-event distances. We can see that NB has the best performance in terms of communication overhead, and has almost comparable overlapped area as the greedy centralized approach. Furthermore, one good advantage of our approach is that the communication overhead does not increase as the sink-to-event distance increases. This is expected since the coordination of sensors and actors are done without any involvement from the sink. For the approaches with sink involved, like the CSC and MDS, the processing of commands incur much higher overhead when the distance from the sink to event region increases due to the overhead incurred in reporting the event from the sensors in the event region to sink, and the commands issued to actors in the actor set from the sink.

7 Correlation Aware Data Gathering in Wireless Sensor Networks

7.1 Introduction

Wireless Sensor Networks (WSNs) have gained tremendous importance in recent years because of their potential use in various fields. The devices used for sensing and communication in such networks are usually small, cheap and low powered and hence, have limited resources for computation as well as communication. This has spurred a need for energy efficient protocols tailored specifically toward sensor network environments.

One of the key tasks performed by any WSN is the collection of sensor data from the sensors in the field to

the sink for processing. This task is also referred to as *data gathering*. In this work, we consider the problem of data gathering in environments where the data from the different sensors are correlated. Such correlation of the data being collected can be leveraged by appropriately fusing the data inside the network to the best extent possible, thereby reducing the number of transmissions and hence energy consumption, for the gathering process.

On the other hand, a data gathering tree that does not explicitly make use of the correlation between sensor data can be considered to be correlation unaware. The most representative structure for correlation unaware aggregation approaches is a *Shortest Path Tree* (SPT). Since the primary goal of the structure is to minimize delay, SPT is not considered to be a correlation-aware data gathering structure. Even though opportunistic aggregation may possibly occur when different paths overlap with each other, it does not necessarily maximize the degree of aggregation possible in the network.

The objective of correlation aware data gathering is to reduce the energy cost of an aggregation tree. The energy optimal aggregation structure for a data gathering application depends on the degree of correlation existing between the source data. For statistical queries such as min, max, avg, etc., two pieces of data can be combined and reduced to the same size as that of the original pieces. We call this type of correlation as *perfect* correlation. It is well-known that when sensor data are perfectly correlated, the *Steiner Minimum Tree* (SMT) over all the sources, sink and some of the non-source nodes is optimal. On the other hand, there are other scenarios where the message sizes may not be reduced to the same size as the original data; only a part of each piece of information is redundant. If the correlations between sensor data are not perfect, there is no established optimal structure. Hence, several attempts ([12], [42]) have been made to propose heuristics to approximate the optimal solution.

In this work, we investigate the energy efficiency of the correlation aware aggregation process through comprehensive quantitative analysis and leverage the insights gained to design a distributed and highly efficient aggregation solution. We specifically explore how the improvement in energy efficiency is impacted by network conditions, defined by several parameters including the node density, source density, the physical distribution of sources, the correlation degree, and the delay bound. We present observations from the simulation results, and draw inferences on the trade-offs involved in achieving energy efficiency.

In studying the improvements in energy efficiency with respect to specific network parameters, we also answer two fundamental questions:

1. *Is there a practical limit on the achievable improvement in energy efficiency by adopting a correlation aware aggregation structure as opposed to a correlation unaware structure?* The answer to this question will establish practical bounds on the energy efficiency improvement that can be achieved, and in turn provide a motivation or lack there-of for performing correlation aware aggregation in the first place.

2. *Is there a maximum usable delay bound that can deliver the maximum achievable energy cost improvement?* The answer to this question will establish a practical bound on how delay tolerant a WSN application needs to be in order to get the maximum energy efficiency benefit.

Building on the results of analysis, we present a simple, scalable, and distributed approach called *SCT* (Semantic/Spatial Correlation-aware Tree) that does not require any centralized coordination while still achieving potential cost benefits due to efficient aggregation. The SCT structure is instantaneously constructed during the course of a single query delivery and does not require any knowledge of the number of sources or their locations. The SCT approach, with its highly manageable structure, ensures low maintenance overhead of the aggregation structure, eliminates the need for global synchronization among sensors for aggregation of sensor-data, while also addressing the other challenges such as load balancing and node failures. Through simulations and analysis, we establish the message costs incurred by SCT for a variety of network conditions, and compare them with an ideal correlation-aware and a correlation unaware approach. We show that SCT, though simple in its realization, can achieve substantial performance benefits.

Our contributions can thus be summarized as follows:

- We characterize through quantitative analysis how the energy improvement of a correlation aware aggregation structure is impacted by different network parameters. We show that the energy improvement tends to be bounded by a small constant under many network scenarios. Furthermore, the improvement corresponds to when the additional cost of establishing a correlation aware structure is not taken into account, in the presence of which the improvement will be further reduced.
- We also characterize what the maximum usable delay bound is for achieving the maximum energy efficient structure. We show that the maximum usable delay bound is a small constant times the delay along the maximum length shortest-path in the default shortest path tree.
- We use the insights obtained to design a simple.scalable, distributed correlation approach called SCT (Semantic/Spatial Correlation Tree) and evaluate its benefits using simulation.

7.2 Analysis of Bounds on Improvements through Correlation Aware Data Gathering

In this section, the motivation for the approach called SCT is described. Specifically, we are interested in answering the following questions

1. *Is there a practical limit on the achievable improvement in energy efficiency by adopting a correlation aware aggregation structure as opposed to a correlation unaware structure?*
2. *Is there a maximum usable delay bound that can deliver the maximum achievable energy cost improvement?*

In this context , the study is carried out using both qualitative and quantitative analysis.

7.2.1 Model, metrics and algorithms

We use a custom-built simulator written in C++ for all our simulations. We consider the aggregation tree cost - the number of edges on a given aggregation tree - as the measure of energy efficiency of the corresponding data gathering process. The metric we use to measure the energy efficiency improvement provided by correlation aware trees is the *cost ratio*, which is defined as the ratio of the cost of the correlation unaware tree to that of the correlation aware tree over the same set of sources and sink. We consider a typical sensor network scenario where a total of n sensors are randomly distributed in a disk of radius R. Of the n sensors, k are randomly chosen as sources to report data to the sink, which is located at the center of the disk. The network parameters are used for the evaluation are: Delay bound, Node density, Source denisty,Source distribution and Correlation Degree. We choose Steiner Minimum Tree (SMT) as the correlation aware structure, since it is the optimal aggregation structure [12] when sensor data are perfectly correlated. On the other hand, SPT is selected as the correlation unaware structure since it minimizes the delay required for data aggregation. To evaluate the impact of delay sensitivity of the application on the cost of a near-optimal tree, we need an algorithm that generates near-optimal trees for various delay constraints. Specifically, if the delay bound for a certain data gathering task is D, the delay incurred on the longest path of the near-optimal aggregation tree should be less than or equal to D. From hereon, *we refer to the delay-bounded near-optimal tree as DB-SMT (delay-bounded steiner minimum tree) and the near-optimal tree without delay bound as simply the SMT.* We choose an algorithm called BSMA (bounded shortest multicast algorithm) to generate the DB-MST.

7.2.2 Qualitative Analysis

In this section, we theoretically substantiate the slow rate of growth of the cost ratio of SPT over SMT with respect to node density. Specifically, we show that the expected (energy) cost improvement obtained by a SMT over SPT scales very slowly (as $\sqrt{\log n}$) with node density.

We consider a network graph where nodes are uniformly distributed in a unit area disk and the root of the SPT tree is at the center of the disk. For the convenience of analysis, we divide the network into layers of concentric rings, each ring consisting of all the nodes that are at the same distance (in terms of hops) away from the sink, i.e. nodes in between the i^{th} and $(i-1)^{th}$ rings are assumed to be i hops away from the sink. The distribution of nodes and sources are assumed to be uniform in the network. In a SPT structure, each source is connected to the sink located at the center of the unit disk. For sources further away from the sink, the shortest paths can be considered to be independent of each other with a high probability. However, at a certain distance away from the sink, all shortest paths tend to converge, and the nodes within this distance belong to at least one of the shortest paths with a high probability. Hence, we assume that there exists a threshold distance and hence ring i^* exists, such that for all $i \leq i^*$, all nodes on ith ring are part of the SPT structure. However, for rings beyond ring i^*, only some of the nodes on each ring will be part of the SPT structure.

Based on the SPT structure defined above, we define a relaxed SPT structure SPT_r, such that the cost of SPT_r is higher than that of SPT. Now, let m be the hop number of longest shortest path in the network, n and

s be the total number of nodes and sources in the network respectively. The expected cost of the relaxed SPT structure is now given by,

$$E[C_{sptr}] = \frac{i^{*2}}{m^2}n + \frac{2}{3}\frac{s}{m^2}(m - i^*)(m^2 - i^{*2}) \tag{6}$$

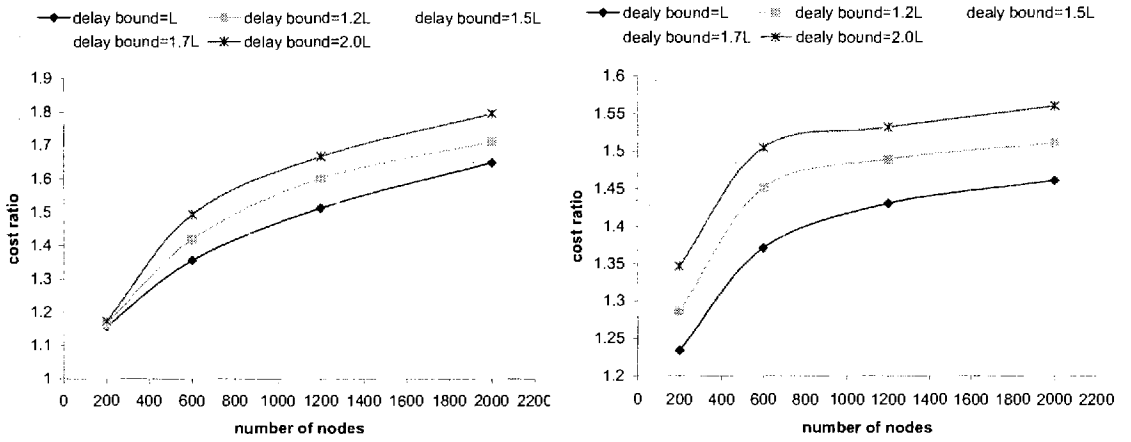
where m is given by,

$$m = 1.32\sqrt{\frac{n}{\log 10n}} \tag{7}$$

The expected cost improvement of SMT over SPT in sensor network graph increases at $\Theta(\sqrt{\log n})$, where n is the total number of node in the sensor network, and s is $\Theta(n)$.

7.2.3 Quantitative Analysis

In this section we present simulation results to show the energy-delay tradeoffs of aggregation trees under various network conditions.



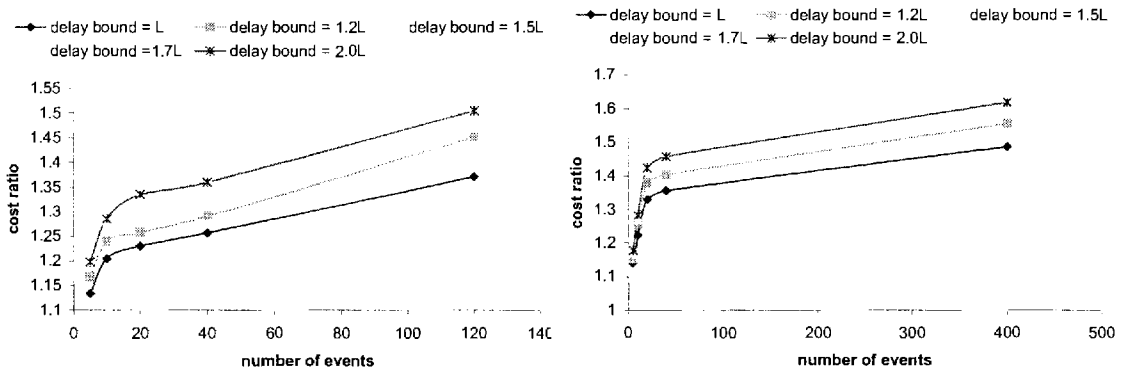
(a) *Source density* $= 1/20$ (b) *Source density* $= 1/5$

Figure 17: Performance Improvement over SPT for Different Node Densities

7.2.4 Varying Node Density

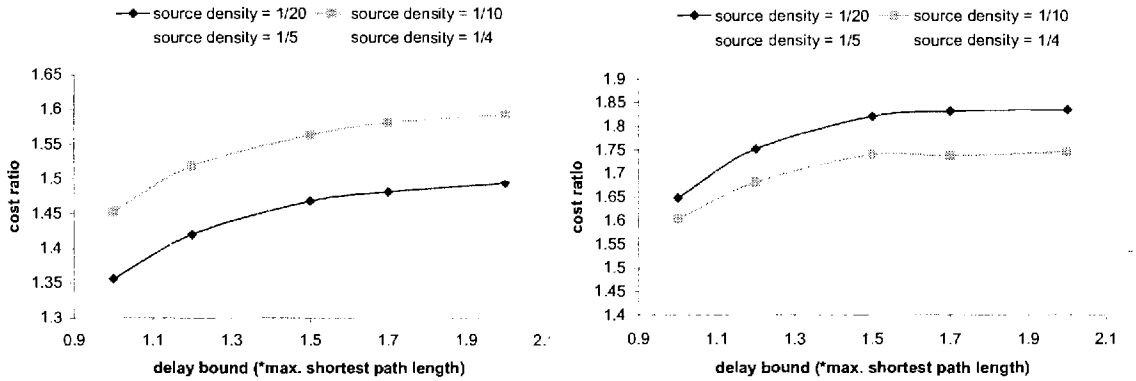
To study the impact of node density on the energy efficiency of aggregation trees, the number of sensor nodes distributed in the field (n) is increased from 200 to 2000. Figure 17 shows the cost ratio of SPT vs. DB-SMT when the source densities are $1/20$ and $1/5$.

It can be observed from the results that the cost ratio between SPT and DB-SMT increases with node density. This implies that correlation aware data gathering is more efficient when the density of sensor nodes is large. This can be intuitively explained as follows: with high node density, the probability of shortest paths overlapping with each other is low; hence, SPT has very low aggregation efficiency and there is greater potential for energy improvement using a DB-SMT. Consequently, the cost ratio improves as node density increases.



(a) $n = 600$ (b) $n = 2000$

Figure 18: Performance Improvement over SPT for Different Source Distribution



(a) $n = 600$ (b) $n = 2000$

Figure 19: Performance Improvement over SPT for Different delay bounds

7.2.5 Varying source distribution

In this set of simulations, n increases from 200 to 2000, and each configuration has a total of $s = n/5$ sources distributed in the network. The number of events e (locations) in the network for each scenario increases from 5, 10, 20, 40 to s. From the results presented in Figure 18, it can be seen that the cost ratio increases with the number of events.The sources tend to be densely distributed around event locations when there are few events in the network.Hence, the shortest paths from the same event location to the sink can combine with each other at an early stage, thereby making SPTs inherently efficient in terms of path sharing.However, the path diversity of SPTs tends to increase as the number of event locations increases with the source distribution tending towards uniform distribution.

7.2.6 Varying delay bounds

To study the variation of cost ratio with respect to delay bounds in depth, results plotted in Figure 19.It can be clearly seen that the cost ratio increases with increasing delay bounds, which indicates that less restrictive delay tolerance helps improve the aggregation and hence the cost efficiency. Higher delay bounds imply that the aggregation path can be longer in order to maximize en-route aggregation. However, simulation results show that aggregation paths longer than twice the longest shortest path do not help significantly in reducing the cost.

Similarly, the trends for other parameters like source density and correlation degree were also obtained.

7.2.7 Summary

In this subsection, we summarize all the observations and insights derived .

- *The cost ratio of SPT over DB-SMT scales very slowly (tends to saturate) with respect to node density.*
- Further, when node density is high, the cost ratio of SPT over DB-SMT decreases with increasing source density. However, at low node density, a moderate source density delivers the best cost improvement.
- For different correlation models, the energy efficiency of DB-SMT increases with correlation degree, and DB-SMT with the lowest delay bound is the most energy efficient for low to moderate correlation degrees. Higher delay bounds help improve aggregation efficiency only when correlation degree ρ is sufficiently high. The high correlation degree also ensures the optimality of SMT.
- Most importantly, the energy delay tradeoff of correlation aware and unaware tree can be summarized as follows: The cost ratio of SPT over DB-SMT increases as delay bound increases for high correlation degrees. *Delay bounds beyond twice the maximum shortest path length do not help reduce DB-SMT cost further.* Furthermore, the cost ratio tends to decrease as delay bound increases for low correlation degrees.

7.3 Problem statement and challenges

7.3.1 Problem Definition

We consider a multi-hop WSN with one sink at the center and n sensors distributed randomly in a circular field according to a Poisson process.[10] The sink sends a query and k of the n sensors respond to the query. We refer to these sensors that have information to send as sources . As a measure of the energy efficiency of a data gathering structure, we define its *message cost* as the total number of transmissions required for responses from

[10]Note that the assumptions about the shape of the sensor field and the location of the sink are made for better illustration of the proposed approach and are not essential to the solution.

all k sources to reach the sink. Our primary goal is to minimize message cost when there is correlation present between data from different sources.

The following two types of correlations are considered in this paper:

- *Spatial Correlation:* This refers to the correlation of the data reported by multiple sensors sensing the same event or phenomenon. For example, consider the query: What is the temperature in the region defined by the rectangle (x1,y1,x2,y2)? Given the typical dense deployments of sensors in WSNs, it is likely that the sensing regions of two different sensors within the rectangular region overlap and the data reported are spatially correlated.
- *Semantic Correlation:* This refers to the correlation of data reported by multiple sensors due to the semantics of the query. For example, consider the query: *Is the total number of cars in the rectangular region (x1,y1,x2,y2) greater than K?* In this case, even if the sensors are reporting data about *different* cars, the information reported is correlated as it is only required to find the total number of cars and consequently determine if it is greater than K.

7.3.2 Challenges

There are several challenges that must be tackled inorder to realize an energy efficient correlation approach. The most important being:

- Construction: The foremost consideration in building an aggregation structure is the manner in which the aggregation structure is constructed. A desirable practical solution should consider the tradeoffs between the overhead involved in the construction process itself on the one hand, and the message cost of the aggregation structure on the other hand, and ensure that it is reasonably efficient across all query and response paradigms.
- Maintenance: An aggregation structure may be modified or reconstructed after a certain period of time to accommodate load balancing, node failures or for any other reasons.
- Synchronization: One of the main considerations for any aggregation scheme is the time each node has to wait before it aggregates the messages received from all sources downstream of it. In the absence of such timing requirements,messages from some downstream sources may arrive after aggregation at a particular aggregation node and hence need to be transmitted separately (consequently increasing the message cost).
- Handling variability: An aggregation approach should also be reasonably efficient in terms of message cost when the degree of correlation varies ($0 < \rho \leq 1$). In addition, it should be able to perform efficient aggregation irrespective of the distribution of sources.

7.4 SCT Design and Approach

The design of SCT is predicated on two important elements:

- An aggregation backbone facilitating the generation of efficient aggregation trees
- A fixed structure independent of source distribution and density.[11]

These two design elements address the challenge of efficient construction, and incorporate the characteristics and requirements of sensor networks. The SCT approach is described below in detail. We present the different phases of the data aggregation process as well as provide insights for the design choices.

7.4.1 Division of the Network

During the setup phase, the sink propagates the following information to the entire network: (i) location of itself, (X_s, Y_s), (ii) the total number of nodes, n (iii) the radius of the network, R and (iv) the computed values for m and n_0 to all the nodes in the network. Each node in the network is assumed to know it's own geographical location. When a node receives the packet, it first computes the distance between itself and the sink. This determines the ring, i, to which it belongs to. For example, any node at a distance d from the sink, such that $(i-1)\frac{R}{m} < d \leq i\frac{R}{m}$, belongs to the ith ring. Each node can calculate the number of sectors per ring $s(i)$ as: $s(i) = \lceil \frac{2(i-1)n}{n_0 m^2} \rceil$. Given the locations of the node with respect to the sink and the number of sectors within the ring, any node can determine the sector number to which it belongs.

7.4.2 Determination of Aggregation Nodes

In the SCT approach, the aggregation nodes are selected by leveraging the fixed, geometric division of the sensor field. For each sector in a given ring, the geometric center of the lower arc bounding the sector is defined as the ideal location of the aggregation node for this sector. The node closest to this ideal location is chosen as the aggregation node. Given the value of m and n_0, each source not only knows the sector and ring numbers to which it belongs but can also determine the boundaries of the sector. If we are to adopt polar coordinates and if α and β are the bounding angles of a sector corresponding to the ith ring, the location of the aggregation node is given by $((i-1)\frac{R}{m}, \frac{\alpha+\beta}{2})$.

7.4.3 Event-driven Data Collection

To achieve maximum aggregation of the source data at the aggregation nodes, it is also necessary to ensure that these nodes wait for an optimum delay value. In SCT we use a more desirable alternative where there are only coarse-grained timers and the aggregation process is mainly event-driven. This approach is motivated by

[11] ratio of the number of sensors that send data packets to the sink to the total number of sensor nodes in the network

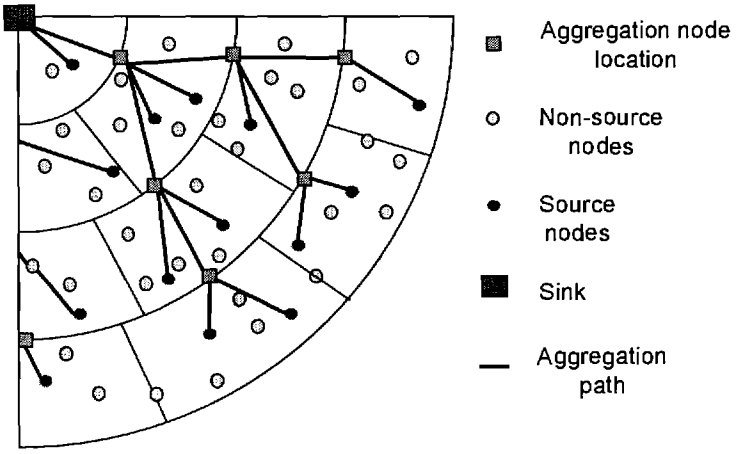


Figure 20: A Subsection of the SCT Aggregation Structure

the fact that each aggregation node knows the exact number of children that are also aggregation nodes. When an aggregation node receives information from all children that are aggregation nodes, it is assured that the data from all sources within the sector are also received by an aggregation node. This is because the sources transmit their data at the beginning of each message collection round while the aggregation nodes wait for the notifications from all the downstream aggregation nodes. The arrival of messages from all downstream aggregation nodes is used as the trigger to merge and propagate the information collected, upstream towards the sink. Thus, synchronization is achieved in an event-driven fashion without the need for explicit delay timers at each aggregation node.

7.4.4 Load Balancing

We propose two simple schemes to distribute the roles of aggregation nodes to different sets of nodes over a certain period of time:

1. *Location of the rings*: In the current SCT description, the different rings are of width $\frac{R}{m}$, where R is the radius of the network and m is number of rings. To do load balancing, the location of the first ring can be shifted by a distance $\frac{R}{m} - rc$, where r is the one-hop transmission range and c is a small integer that is varied from $0 \ldots \frac{R}{mr}$. The same offset is applied to every ring so that the width of the ring is still maintained to be $\frac{R}{m}$ for all rings except the first and last.

2. *Orientation of the sectors*: In a similar way, we can choose the offset angle for a sector to be different across multiple queries. The offset angle, θ, can be incremented according to the relation, $\theta = \frac{c}{s(i)}$ where $s(i)$ is the number of sectors in the ith ring and c is a small integer dependent on the query identifier. This again assures that different nodes are chosen as aggregation nodes over several query floods.

7.4.5 Aggregation Reliability and Node Mobility

The failure of any non-aggregation node will not impact the correctness or efficiency of the SCT approach. Therefore, here we only discuss how the aggregation node failures are addressed in SCT. Aggregation node failures before the setup of the SCT structure can be identified by the lack of announcement from the particular node, and another node closest to the ideal location can announce itself as the aggregation node of this sector. If an aggregation node fails during the information collection phase, the lack of ACK messages from this node to sources can inform them of its failure. In this case, the retransmission of the first packet enables the election of a new aggregation node, which in turn broadcasts an announcement upon receiving the retransmitted message. After the re-election, aggregation proceeds as usual.

In the case of node mobility, the proposed solution needs to be modified to accommodate mobility in (i) sources, (ii) aggregation nodes and (iii) other nodes. If the sources are mobile but the aggregation nodes are fixed, sources will forward to the closest aggregation node given its current location. If aggregation nodes are mobile, the node failure handling mechanism can be leveraged to elect a new aggregation node for that sector. Mobility of other nodes does not affect the SCT approach.

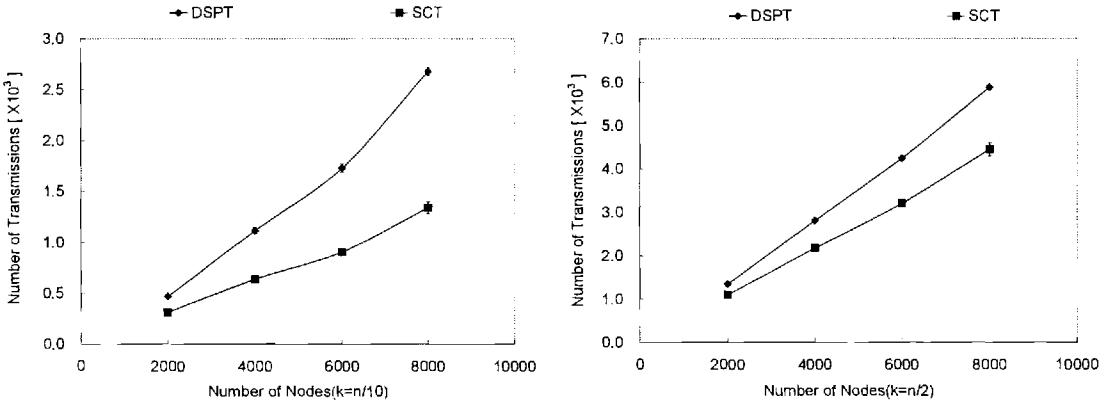
7.5 Performance Evaluation

In this section we evaluate the performance of the SCT approach under different network configurations and compare it with two centralized schemes: minimum Steiner Tree, SPT, and one decentralized scheme: DSPT (Decentralized Shortest Path Tree). We vary the node density, source density, source distribution, as well as correlation coefficient (ρ) and evaluate the message cost of the four structures under different scenarios.

We use a discrete event simulator based on the LECS simulator for all evaluations. The simulation topologies are largely similar to those used in general sensor networks: 2000 to 8000 nodes are uniformly distributed within a circular field of radius 400m. The number of sources that generate messages for one specific query varies from $\frac{1}{10}$, $\frac{1}{6}$, $\frac{1}{4}$ to $\frac{1}{2}$ of the total number of nodes in the network. We evaluate the SCT approach using two metrics: message cost and data gathering latency. For message cost, we measure the total number of transmissions required for all responses to reach the sink for one round of data collection, and for data gathering latency, we measure the time interval from the time when all sources start to send messages, to the last message reaches the sink. To highlight the benefit of SCT as a distributed solution, a decentralized version of the shortest path tree (DSPT) is also included in the evaluation. In DSPT, GPSR routing protocol is used to approximate SPT in a distributed fashion.

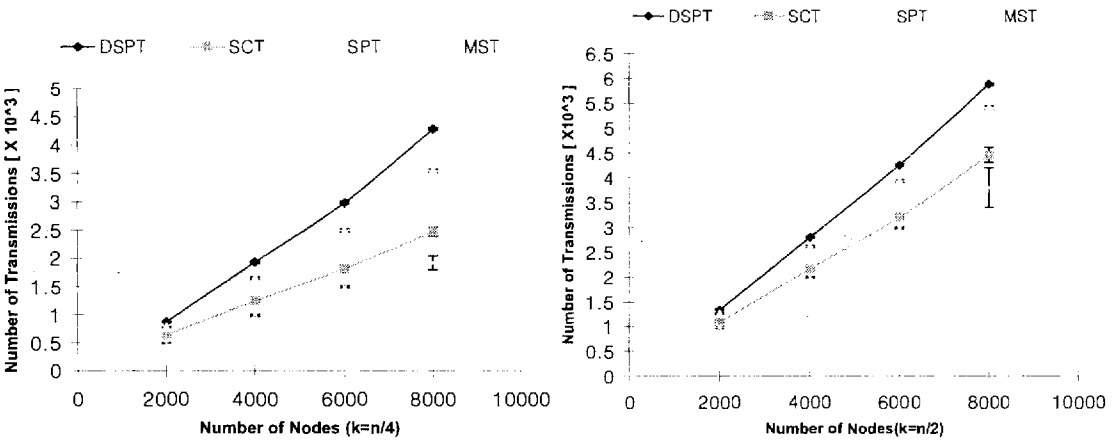
7.5.1 Perfect correlation results

Figures 21 (a), (b), and (c) show the cost of two decentralized schemes as a function of the number of nodes for different numbers of sources k. It can be seen that SCT outperforms DSPT scheme under all situations. Interestingly, we observe that the cost of DSPT is up to 200% of the SCT cost as the number of nodes increases.



(a) $k = n/10$ (b) $k = n/2$

Figure 21: Performance comparison between SCT and DSPT: Perfect correlation



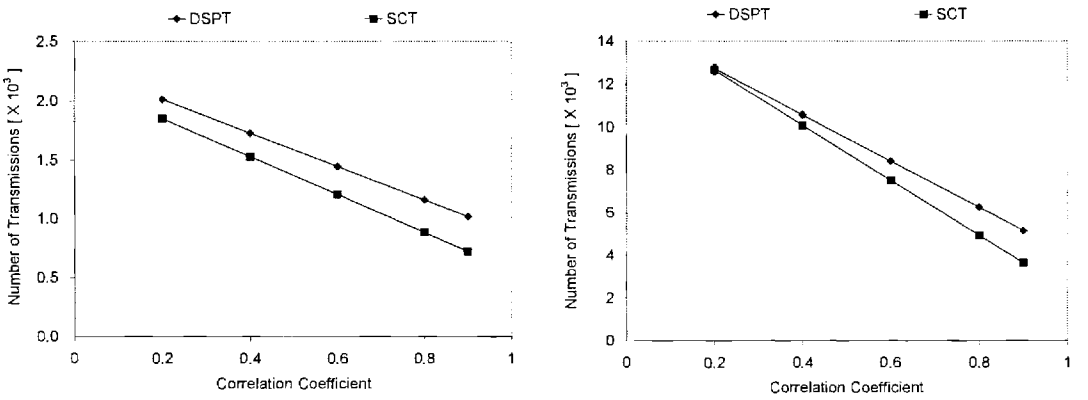
(a) $k = n/4$ (b) $k = n/2$

Figure 22: Performance Comparison between SCT and Centralized Schemes, SPT and MST

The DSPT cost also increases faster than the SCT cost as node number increases. This is expected since more nodes reduces the efficiency of aggregation in DSPT as the paths chosen by different sources are less likely to overlap. Therefore, SCT can be considered a more scalable approach.

7.5.2 Decentralized vs Centralized Schemes

We compare the performance of the decentralized SCT and DSPT schemes with the centralized schemes they approximate. Figure 22 shows the cost of the proposed scheme and the centralized schemes as a function of the node number. To evaluate the cost of the centralized schemes, we assume perfect aggregation for both SPT and MST. From the figures we can see that DSPT's message cost approaches closely that of SPT while SCT's message cost approaches closely the cost of MST. Furthermore, although SCT is a decentralized scheme



(a) $n = 2000, k = 500$ (b) $n = 8000, k = 2000$

Figure 23: Performance Comparison between SCT and DSPT for varying correlation degree ρ

without perfect aggregation, it still outperforms the centralized SPT, since SCT explicitly aggregates sensor data, while SPT just leverages aggregation opportunistically.

7.5.3 Correlation Degree

In Figure 23, we characterize the message complexity of SCT when the correlation coefficient varies from 0.2 to 0.9. Notice that in this graph, the number of transmissions is normalized to a unit message size. For example, if after aggregation, a node transmits a message of size 1.5 times the unit message size, it is counted as 1.5 transmissions. From this figure, we can see that for both DSPT and SCT, message cost reduces as ρ increases, since the two schemes have either implicit or explicit mechanisms to leverage the correlation. However, the message cost of SCT reduces faster than DSPT because it facilitates aggregation at an earlier stage of packet forwarding, hence can reduce packet transmission cost more effectively.

7.5.4 Other results

The approach was analyzed for localized events and also delay bounds. In both cases SCT outperforms DSPT since its message cost does not grow fast unlike DSPT.

7.6 Related Work

There are a number of Correlation-Unaware routing approaches proposed for sensor networks and most of these use one of the two popular choices for their design: Protocols such as directed diffusion [27] use query paths to construct sensors-to-sink routes, while protocols such as GPSR [30] use location information of sensors and the sink to forward messages to the sink. Also there are several Correlation-Aware routing approaches that use

explicit aggregation strategies. It is known that under conditions of complete knowledge of source locations a Steiner tree over all sources, sink and non-source nodes gives the optimal message cost. However computing the Steiner tree has been shown to be NP-hard [51].and several simple heuristics based approaches to find an approximate Steiner tree have been proposed in [12, 13]. [29] and [21] address the more general problem of building aggregation structure with optimal expected cost when the knowledge of sources is incomplete.

There are a number of works that look at the efficiencies of the various data aggreagation trees. For example [32] systematically compares data-centric routing approaches in wireless sensor networks. However, the focus of [32] is on comparing data-centric routing with traditional end-to-end routing scheme(address-centric routing).The emphasis of [32] is to compare the performance differences between "aggregate" and "do not aggregate", while our work investigates energy cost differences between aggregation aware and unaware schemes (SPT with aggregation and DB-SMT tree). [42] compares two major classes of data aggregation scheme: routing-driven compression (RDC) and compression-driven routing (CDR) across a broad range of spatial correlations. This work mainly investigate the impact of correlation degrees on optimal aggregation structure. While for our study, we consider not only correlation degrees, but delay bounds and other parameters when comparing efficiency of correlation aware and unaware data aggregation trees.

7.7 Conclusion

In the first part of this work, the PIs study the energy efficiency of correlation aware aggregation trees in wireless sensor networks. Sensor applications with and without delay tolerance are considered, and how delay tolerance and other network conditions affect the efficiency of an correlation aware aggregation tree is explored. Through quantitative study and analysis, we conclude two rather surprising results: the energy improvement in using correlation aware aggregation is not significant under many network scenarios compared to the cost and complexity incurred in the tree construction process; and the maximum useable delay bound required to achieve the best possible energy efficiency is not high compared with the delay along the maximum length shortest-path in the default shortest path tree. Practical implications of these results have also been identified. In the second part, we propose a novel solution to aggregate correlated information from a subset of sensors to the sink using the results of the analysis performed in the first part. The proposed scheme is scalable, distributed, requires minimal computation and is highly-manageable compared with existing solutions. The proposed scheme is assessed both intuitively and analytically. Through simulations, we compared the proposed scheme with ideal, centralized data structures as well as a distributed structure. Simulation results show that as a correlation-aware structure, SCT performs significantly better than correlation-unaware structures in terms of message cost and data gathering latency.

8 Publications Resulted from This Project

Journal Publications:

- Tommaso Melodia, Dario Pompili, I.F. Akyildiz, "Handling Mobility in Wireless Sensor and Actor Networks," submitted to IEEE Transactions on Mobile Computing, 2007.
- Tommaso Melodia, Dario Pompili, Vehbi C. Gungor, and Ian F. Akyildiz. "Communication and Coordination in Wireless Sensor and Actor Networks," *IEEE Transactions on Mobile Computing*, Vol. 6, No. 10, pp. 1116-1129, October 2007.
- V. C. Gungor, O. B. Akan, I. F. Akyildiz, "A Real-Time and Reliable Transport Protocol for Wireless Sensor and Actor Networks," *to appear in IEEE/ACM Transactions on Networking for publication*, 2008.
- V.C. Gungor, M.C. Vuran and O.B. Akan, "Analysis of Congestion and Contention in Wireless Sensor and Actor Networks," Ad Hoc Networks Journal (Elsevier), September 2007.
- R. Vedantham, Z. Zhuang, and R. Sivakumar. "Addressing Hazards in Wireless Sensor and Actor Networks," In *Mobile Computing and Communications Review (MC2R)*, 2006.
- R. Vedantham, Z. Zhuang, and R. Sivakumar. "Hazard Avoidance in Wireless Sensor and Actor Networks, " In *Elsevier Computer Communications Journal, Special Issue on Wireless Sensor Networks*, 2006.

Conference Publications:

- T. Melodia, D. Pompili, V. C. Gungor, and I. F. Akyildiz. "A Distributed Coordination Framework for Wireless Sensor and Actor Networks," In *Proceedings of ACM Mobihoc 2005*, Urbana-Champaign, IL, May 2005.
- T. Melodia, D. Pompili, and I. F. Akyildiz. "A Communication Architecture for Wireless Sensor and Actor Networks," In *Proc. of IEEE Intl. Conf. on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, Reston, VA, September 2006.
- R. Vedantham, Z. Zhuang, and R. Sivakumar. "Addressing Hazards in Wireless Sensor and Actor Networks," In *ACM International Conference on Mobile Computing and Networking (MOBICOM) (Poster Presentation)*, August 2005.
- R. Vedantham, Z. Zhuang, and R. Sivakumar. "Hazard Avoidance in Wireless Sensor and Actor Networks," In *IEEE International Conference on Broadband Networks (BROADNETS)*, October 2005.
- R. Vedantham, Z. Zhuang, and R. Sivakumar. "Mutual Exclusion in Wireless Sensor and Actor Networks," In *ACM International Conference on Mobile Computing and Networking (MOBICOM) (Poster Presentation)*, August 2005.

- S.-J. Park and R. Sivakumar, "Sink-to-Sensors Reliability in Sensor Networks" Extended Abstract in *Proceedings of ACM MOBIHOC*, Annapolis, MD USA, June 2003.
- S.-J. Park, R. Vedantham, R. Sivakumar and I.F. Akyildiz, "A Scalable Approach to Reliable Downstream Data Delivery in Wireless Sensor Networks," Proceedings of *ACM MOBIHOC*, Roppongi, Japan, May 2004.
- R. Vedantham, S.-J. Park and R. Sivakumar, "Sink-to-Sensors Congestion Control," in *IEEE International Conference on Communications* (ICC), Seoul, Korea, May 2005.
- Y. Zhu, K. Sundaresan and R. Sivakumar, "Practical Limits on Achievable Energy Improvements and Useable Delay Tolerance in Correlation Aware Data Gathering in Wireless Sensor Networks," *IEEE Conference on Sensor and Ad Hoc Communications and Networks* (SECON), Santa Clara, California, USA, September 2005 (Best Paper Award)
- Y. Zhu, R. Vedantham, S.-J. Park and R. Sivakumar, "A Scalable Correlation Aware Aggregation Strategy for Wireless Sensor Networks," *IEEE International Conference on Wireless Internet* (WICON), Budapest, Hungary, July 2005.
- Y. Zhu and R. Sivakumar, "Challenges: Communication through Silence in Wireless Sensor Networks," *ACM MOBICOM*, Cologne, Germany, August 2005.

References

[1] CPLEX solver.

[2] The J-Sim Simulator.

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

[4] O. Akan and Ian F. Akyildiz. Event-to-Sink Reliable Transport in Wireless Sensor Networks. *IEEE/ACM Transactions on Networking*, 13(5):1003–1017, October 2005.

[5] I. F. Akyildiz and I. H. Kasimoglu. Wireless sensor and actor networks: Research challenges. *Ad Hoc Networks (Elsevier)*, 2(4):351–367, October 2004.

[6] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Computer Networks (Elsevier)*, 38(4):393–422, March 2002.

[7] F. Aurenhammer. Voronoi Diagrams - A Survey Of A Fundamental Geometric Data Structure. *ACM Computing Surveys*, 23:345–405, 1991.

[8] Robert Grover Brown and Patrick Y. C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering, 3rd edition.* John Wiley & Sons, Inc., Hoboken, NJ, 1996.

[9] L. Krishnamurthy C-Y. Wan, A. T. Campbell. Psfq: a reliable transport protocol for wireless sensor networks. *Proceedings of WSNA*, September 2002.

[10] Y. Uny Cao, Alex S. Fukunaga, and Andrew B. Kahng. Cooperative Mobile Robotics: Antecedents and Directions. *Autonomous Robots*, 4:1–23, 1997.

[11] George Coulouris, Jean Dollimore, and Tim Kindberg. *Distributed Systems: Concepts and Design.* Addison-Wesley, 2001.

[12] R. Cristescu, B. Beferull-Lozano, and M. Vetterli. On Network Correlated Data Gathering. In *INFOCOM*, Hong Kong, March 2004.

[13] R. Cristescu and M. Vetterli. Power Efficient Gathering of Correlated Data: Optimization,NP-Completeness and Heuristics. In *The Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, June 2003.

[14] J. Czyzyk, M. Mesnier, and J. Mor. The NEOS server. *IEEE Journal on Computational Science and Engineering*, 5:68–75, 1998.

[15] Saumitra M. Das, Himabindu Pucha, and Y. Charlie Hu. Performance Comparison of Scalable Location Services for Geographic Ad Hoc Routing. In *Proceedings of IEEE INFOCOM 2005*, Miami, FL, USA, March 2005.

[16] M. Caccamo et al. An Implicit Prioritized Access Protocol for Wireless Sensor Networks. In *Proceedings of IEEE Real-Time Systems Symposium*, Texas, USA, December 2002.

[17] Emad Felemban, Chang-Gun Lee, Eylem Ekici, Ryan Boder, and Serdar Vural. Probabilistic QoS Guarantee in Reliability and Timeliness Domains in Wireless Sensor Networks. In *Proceedings of IEEE INFOCOM 2005*, Miami, FL, USA, March 2005.

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

[19] Brian P. Gerkey and Maja J. Mataric. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *The International Journal of Robotics Research*, 23(9):939–954, September 2004.

[20] V. C. Gungor, O. B. Akan, and I. F. Akyildiz. A Real-Time and Reliable Transport Protocol for Wireless Sensor and Actor Networks. *submitted to IEEE/ACM Transactions on Networking for publication.*

[21] A. Gupta, M. Pal, R. Ravi, and A. Sinha. Boosted Sampling: Approximation Algorithms for Stochastic Optimization. In *Proceedings of The 34th Annual ACM Symposium on Theory of Computing (STOC'04)*, pages 417 – 426, Chicago, IL, 2004.

[22] H. Gupta, S. Das, and Q. Gu. Connected Sensor Cover: SelfOrganization of Sensor Networks for Efficient Query Execution. In *Proceedings of the ACM symposium on Mobile Ad Hoc Networking and Computing (ACM MOBIHOC)*, June 2003.

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

[24] J.L. Hennessy and D.A. Patterson. *Computer Architecture - A Quantitative Approach, Third Edition.* Morgan Kaufmann, 2002.

[25] G. Holland and N.H. Vaidya. Analysis of TCP Performance over Mobile Ad Hoc Networks. In *Proceedings of ACM MOBICOM 1999*, Seattle, WA, USA, 1999.

[26] B. Hull, K. Jamieson, and H. Balakrishnan. Techniques for Mitigating Congestion in Sensor Networks. In *Proceedings of ACM SENSYS 2004*, Baltimore, MD, USA, November 2004.

[27] C. Intanagoniwawat, R. Govindan, and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In *MOBICOM*, Boston, MA, USA, August 2000.

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

[29] D. R. Karger and M. Minkoff. Building Steiner Trees with Incomplete Global Knowledge. In *Proceedings of the 41th Annual IEEE Symposium on Foundations of Computer Science (FOCS'00)*, pages 613 – 623, Redondo Beach, CA, 2000.

[30] B. Karp and H. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *In Mobile Computing and Networking*, 2000.

[31] I. Katzela and M. Nagshineh. Channel Assignment Schemes for Cellular Mobile Telcommunication Systems: A Comprehensive Survey. In *IEEE Personal Communications*, pages 10–31, June 1996.

[32] Bhaskar Krishnamachari, Deborah Estrin, and Stephen B. Wicker. The impact of data aggregation in wireless sensor networks. In *Proceedings of the 22nd International Conference on Distributed Computing Systems*, pages 575–578, 2002.

[33] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. Initializing newly deployed ad hoc and sensor networks. In *Proceedings of ACM MobiCom 2004*, Philadelphia, PA, September 2004.

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

[35] Tommaso Melodia, Dario Pompili, and Ian F. Akyildiz. Handling Mobility in Wireless Sensor and Actor Networks. In *Submitted for publication.*

[36] Tommaso Melodia, Dario Pompili, Vehbi C. Gungor, and Ian F. Akyildiz. A distributed coordination framework for wireless sensor and actor networks. In *Proceedings of ACM Mobihoc 2005*, Urbana-Champaign, IL, May 2005.

[37] Petter Ogren, Edward Fiorelli, and Naomi Ehrich Leonard. Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment. *IEEE Transactions on Automatic Control*, 49:1292–1302, August 2004.

[38] S.J. Park, R. Vedantham, R. Sivakumar, and I.F. Akyildiz. A Scalable Approach for Reliable Downstream Data Delivery in Wireless Sensor Networks. In *Proceedings of the ACM symposium on Mobile Ad Hoc Networking and Computing (ACM MOBIHOC)*, May 2004.

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

[40] Mohammad Rahimi, Hardik Shah, Gaurav Sukhatme, John Heidemann, and Deborah Estrin. Studying the feasibility of energy harvesting in a mobile sensor network. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 19–24, Taipai, Taiwan, May 2003.

[41] Michele Rossi and Michele Zorzi. Cost Efficient Localized Geographical Forwarding Strategies for Wireless Sensor Networks. In *Proceedings of the Tyrrhenian International Workshop on Digital Communications, TIWDC 2005*, Sorrento, Italy, July 2005.

[42] Pattem S., Krishnamachari B., and Govindan R. The impact of spatial correlation on routing with compression in wireless sensor networks. In *International Symposium on Information Processing in Sensor Networks*, pages 28 – 35, April 2004.

[43] Luca Schenato, Songhwai Oh, and Shankar Sastry. Swarm coordination for pursuit evasion games using sensor networks. In *Proceedings of the International Conference on Robotics and Automation*, Barcelona, Spain, 2005.

[44] M. Schwartz. Mobile Wireless Communications. In *Cambridge University Press,*, New York, USA.

[45] K. Seada, M. Zuniga, A. Helmy, and B. Krishnamachari. Energy-Efficient Forwarding Strategies for Geographic Routing in Lossy Wireless Sensor Networks. In *Proceedings of IEEE SECON 2004*, Santa Clara, CA, USA, October 2004.

[46] Rahul Shah, Sumit Roy, Sushant Jain, Waylon Brunette, and Gaetano Borriello. Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks. *Elsevier Ad Hoc Networks Journal*, 1(3):215–233, September 2003.

[47] A. Silberschatz, P.B. Galvin, and G. Gagne. *Operating System Concepts, Sixth Edition.* John Wiley and Sons, Inc., 2001.

[48] 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*, 91(7):1002–1022, 2003.

[49] F. Stann and J. Heidemann. RMST: Reliable data transport in sensor networks. In *Proceedings of IEEE SNPA 2003*, pages 102–112, Anchorage, Alaska, USA, May 2003.

[50] K. Sundaresan, V. Anantharaman, H.-Y. Hsieh, and R. Sivakumar. ATP: A Reliable Transport Protocol for Ad-hoc Networks. In *Proceedings of ACM MOBIHOC 2003*, Annapolis, MD USA, 2003.

[51] H. Takahashi and A. Matsuyama. An Approximate Solution for the Steiner Tree Problem in Graphs. In *Math. Japonica 24*, volume 6, pages 573–577, 1980.

[52] ns-2 The Network Simulator. http://www.isi.edu/nsnam/ns/index.html.

[53] R. Vedantham, Z. Zhuang, and R. Sivakumar. Addressing Hazards in Wireless Sensor and Actor Networks. In *ACM International Conference on Mobile Computing and Networking (MOBICOM) (Poster Presentation)*, August 2005.

[54] R. Vedantham, Z. Zhuang, and R. Sivakumar. Hazard Avoidance in Wireless Sensor and Actor Networks. In *IEEE International Conference on Broadband Networks (BROADNETS)*, October 2005.

[55] R. Vedantham, Z. Zhuang, and R. Sivakumar. Mutual Exclusion in Wireless Sensor and Actor Networks. In *ACM International Conference on Mobile Computing and Networking (MOBICOM) (Poster Presentation)*, August 2005.

[56] R. Vedantham, Z. Zhuang, and R. Sivakumar. Addressing Hazards in Wireless Sensor and Actor Networks. In *Mobile Computing and Communications Review (MC2R)*, 2006.

[57] R. Vedantham, Z. Zhuang, and R. Sivakumar. Hazard Avoidance in Wireless Sensor and Actor Networks. In *Elsevier Computer Communications Journal, Special Issue on Wireless Sensor Networks*, 2006.

[58] M. C. Vuran, O. B. Akan, and I. F. Akyildiz. Spatio-temporal correlation: Theory and applications for wireless sensor networks. *Computer Networks (Elsevier)*, 45(3):245–259, June 2004.

[59] J.E. Walter, J. Welch, and N. Vaidya. A Mutual Exclusion Algorithm for Ad hoc Mobile Networks. In *ACM and Wireless Networks Journal Special Issue on Dialm papers*, 2001.

[60] C. Y. Wan, S. B. Eisenman, and A. T. Campbell. CODA: Congestion detection and avoidance in sensor networks. In *Proceedings of ACM SENSYS 2003*, Los Angeles, CA, USA, November 2003.

[61] G. Wang, G. Cao, T. La Porta, and W. Zhang. Sensor Relocation in Mobile Sensor Networks. In *Proceedings of IEEE INFOCOM 2005*, Miami, Fl, USA, March 2005.

[62] 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.