A Biologically Inspired Architecture for Self-Managing Sensor Networks

Pruet Boonma, Paskorn Champrasert and Junichi Suzuki
Department of Computer Science
University of Massachusetts, Boston
{pruet, paskorn, jxs}@cs.umb.edu

Abstract—*This paper describes a sensor network architecture, called BiSNET, which addresses several key issues in wireless sensor networks such as autonomy, adaptability and self-healing. Based on the observation that various biological systems have developed mechanisms necessary to overcome these issues, BiSNET follows certain biological principles such as decentralization, food gathering/storage and natural selection to design sensor networks. This paper describes and evaluates the biologically-inspired mechanisms in BiSNET. Preliminary simulation results show that BiSNET allows sensor nodes to autonomously adapt their duty cycles for power efficiency and responsiveness of data transmission and to collectively self-heal (i.e., detect and eliminate) false positives in their sensor readings.*

1. Introduction

Wireless sensor networks face several challenges. The first challenge is *autonomy*. Since sensors can be deployed in an unattended area (e.g., forest and ocean) or physically unreachable area (e.g., inside a building wall), they are required to operate with the minimum aid from base stations or human administrators.

The second challenge is *adaptability*. Sensor networks are required to operate through adapting to the environmental changes that sensors monitor [1]. For example, sensors may decrease their duty cycles when there is no significant change in their sensor readings. This results in less power consumption in the sensors. Also, when neighboring sensors report environmental changes, a sensor may draw inference from the reports and increase its duty cycle to be more watchful for a potential local environmental change in the future. This can increase responsiveness of the sensor to transmit its sensor data to a base station.

The third challenge is *self-healing*. Sensor reading usually contains some noises; it may be a false positive due to, for example, malfunction of sensors. Sensors are required to collectively self-heal (i.e., detect and eliminate) false positives in their sensor readings instead of transmitting them to base stations [2]. This can reduce power consumption of sensors because in-sensor data processing incurs much less power consumption than data transmission does [3].

This paper describes and evaluates an architecture for sensor networks, called BiSNET (Biologically-inspired architecture for Sensor NETworks), which addresses the above challenges. BiSNET is motivated by the observation that various biological systems have already developed mechanisms to overcome those challenges [4]. The BiSNET runtime operates atop of TinyOS in each sensor node (Figure 1). It consists of a middleware platform and one or more agents. BiSNET models a platform as a hive and agents as bees. Agents are designed to follow several biological principles such as decentralization, autonomy, food gathering/storage and natural selection. Each agent reads sensor data, and discards or reports it to a base station using biological behaviors such as replication, death and migration. A platform runs on TinyOS and hosts agents. Each platform controls the state of sensor node (e.g., sleep and listen), and provides a set of runtime services that agents use to read sensor data and perform their behaviors.

This paper describes the biologically-inspired mechanisms in BiSNET and evaluates their impacts on the autonomy, adaptability, and self-healing of sensor networks. Simulation results show that BiSNET allows sensor nodes to autonomously adapt their duty cycles for power efficiency, to draw inference on potential environmental changes from sensing activities of neighboring nodes, and to collectively self-heal (i.e., detect and eliminate) false positives in sensor reading.

2. Design Principles for BiSNET Agents

(1) Decentralization: Similar to biological systems (e.g., bee colony), there are no centralized entities in BiSNET to control and coordinate agents. Decentralization allows agents to be scalable and simple by avoiding a single point of performance bottlenecks and failures [5, 6] and by avoiding any central coordination in deploying agents[7].

(2) Autonomy: Similar to biological entities (e.g., bees), agents sense their local environments, and based on the sensed conditions, they autonomously behave without any intervention from/to other agents, base stations and human administrators.

(3) Food gathering and storage: Biological entities strive to seek and consume food for living. For example,

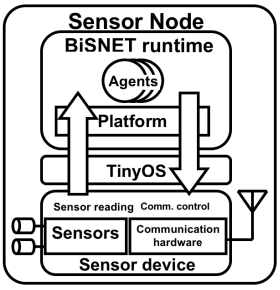

Figure 1. BiSNET Architecture

bees gather nectar from flowers and digest it to produce honey. In BiSNET, agents (bees) read sensor data (nectar) in each duty cycle, and digest it to *energy* (honey)[1]. (Energy gain is proportional to a change between the current and previous sensor data.) They keep some of the energy and deposit the rest in the local platform (hive).

(4) Natural selection: The abundance or scarcity of stored energy in agents affects their behaviors and triggers natural selection. For example, an energy abundance indicates a significant change in sensor reading; thus, an agent replicates itself, and the replicated agent migrates to a neighboring sensor node for reporting sensor data to a base station. An energy scarcity (an indication of few changes in sensor reading) causes the death of agents. Like in biological natural selection where more favorable species in an environment becomes more abundant, the population of agents dynamically changes based on their energy levels (i.e., changes in their sensor readings).

3. Design of BiSNET

This section presents the design of agents and platforms.

3.1. BiSNET Agent

Each agent consists of three parts: *attributes*, *body* and *behaviors*. *Attributes* carry descriptive information regarding the agent, such as energy level, ID/location of a sensor where the agent was born, sensor data to be reported to a base station, and time stamp of the sensor data. Agent designers can add arbitrary data fields as attributes.

Body implements the functionalities of the agent: food gathering and conversion (metabolism) of food to energy. Each agent gathers sensor data (as food) from an underlying sensor device and converts it to energy in each duty cycle.

Behaviors implement actions inherent to all agents. This paper focuses on the following four behaviors.

- *Replication:* Agents may make a copy of themselves as a result of abundance of energy (i.e., indication of a significant change in sensor reading). A replicated (child) agent is placed on the platform that its parent agent resides on, and it receives the half amount of the parent's energy level. Each child agent is intended to move toward a base station to report sensor data.
- *Migration:* Agents may move from one sensor node (platform) to another in response to energy abundance (i.e., indication of a significant change in sensor reading). Migration is used to transmit agents (sensor data) to base stations on a multi-hop and shortest-path basis.
- *Energy exchange:* Agents on a platform always share their energy units (honey) with each other so that their energy levels become equal. A migrating agent shares its energy units with other agents on a destination platform. Also, agents periodically deposit some of their energy units (honey) to their local platforms (hives).
- *Death:* Agents die due to energy starvation when they cannot balance energy gain and expenditure. The death behavior is intended to eliminate agents that carry false positive sensor data. When an agent dies, an underlying platform removes the agent and releases all resources allocated to the agent.

Every agent expends energy to perform migration and replication behaviors. The energy costs for the behaviors are constant for all agents.

Figure 2 shows a sequence of actions that each agent performs in each duty cycle. First, an agent receives sensor data (as nectar) from a sensor device, and converts it to energy (honey). The energy intake (E_F) is calculated with Equation 1. S represents the absolute difference between the current and previous sensor data. M is metabolic rate, which is a constant value between 0 and 1.

$$E_F = \sum_{i}^{L} \frac{S_i}{N} \cdot M_i \quad (1)$$

N is the number of agents running on the local platform. If multiple agents exist on the platform, all of them evenly gain energy (S/N) from sensor data. If a sensor node equips multiple (L) sensor devices (e.g., temperature and humidity sensors), the total energy intake is the sum of energy gains from multiple types of sensor data. Different agents may have different M values to prioritize different sensor nodes. The higher M value an agent has, the more often the agent replicates and migrates because of higher energy intake.

Given E_F, each agent updates its energy level as follows.

$$E(t) = \frac{\sum_i^N E(t-1)}{N} + E_F \quad (2)$$

$E(t)$ is the current energy level of the agent, and $E(t-1)$ is the agent's energy level in the previous duty cycle. t is incremented by one at each duty cycle. Note that agents always exchange and share their energy units equally with other agents in the same platform.

If an agent's energy level ($E(t)$) becomes very low (below death threshold: T_D), the agent dies due to energy star-

[1]The concept of energy in BiSNET does not represent physical battery of sensor node. It is a logical concept that affects agent behaviors.

while $true$
do
- Read sensor data
- Convert sensor data to energy (E_F)
- Update energy level ($E(t)$) through energy exchange
- **if** $E(t) <$ *death threshold* (T_D)
  - **then** Invoke death behavior
- **while** $E(t) >$ *replication threshold* (T_R)
  - **do**
    - Make a child agent.
    - Give the half of energy level to the child agent.
- Deposit energy units (E_P) to the local platform
- **if** *the local platform is in the broadcast state* **and** *# of agents in the local platform* > 1
  - **then** Ask the local platform for migration to a specified sensor node.

Figure 2. Agent Actions in Each Duty Cycle

vation (see also Figures 2 and 3)[2].

An agent replicates itself if its energy level exceeds its replication threshold: T_R (see Figures 2 and 3). The agent keeps replicating itself until its energy level becomes less than its T_R. Agents continuously adjust their replication thresholds as EWMA (Exponentially Weighted Moving Average) of their energy levels:

$$T_R(t) = (1-\alpha)T_R(t-1) + \alpha E(t) \quad (3)$$

$T_R(t)$ is the current replication threshold, and $T_R(t-1)$ is the one in the previous duty cycle. EWMA is used to smooth out short-term minor oscillations in the data series of E. It places more emphasis on the long-term transition trend of E; only significant changes in E have the effects to change T_R. The α value is a constant to control the responsiveness of EWMA against the changes of E.

Each agent deposits a certain amount of energy (E_P) to a platform that it resides on (see also Figures 2 and 3):

$$E_P = \begin{cases} E(t) - E(t-1) & \text{if } E(t) \geq E(t-1) \\ 0 & \text{if } E(t) < E(t-1) \end{cases} \quad (4)$$

Each agent strives to keep its energy level ($E(t)$) close to the one in the previous duty cycle ($E(t-1)$).

When a platform's total energy gain ($\sum E_P$) is greater than a threshold (T_B), the platform changes its state to the broadcast state. This allows agents to migrate to neighboring platforms (see also Figures 2 and 3)[3].

As described above, agents replicate themselves and migrate to neighboring platforms for reporting sensor data to base stations, only when they gain a large amount of energy from a significant change in their sensor readings (i.e., S_i in Equation 1). Agents do not respond to gradual changes in sensor readings (e.g., temperature change during a day or between different seasons). This reduces power consumption in sensor nodes and expands the life of sensor networks.

This adaptive data transmission mechanism is designed with a self-healing capability in mind. When a sensor node does not work properly due to, for example, malfunction of sensors, it generates the agents that carry false positive sensor data. The false positive agents may replicate themselves and migrate to neighboring sensor nodes because the agents' energy levels can be very high. (their energy levels may exceed their replication thresholds (T_R) and the energy gains of their local platforms may exceed the broadcast threshold (T_B)). However, they stop replication shortly because their replication thresholds increase (see Equation 3). They stop migration as well because of energy exchange with other agents at destination platforms, if there is no significant environmental changes at the destinations. Agents can collaboratively eliminate false positive sensor data.

[2]If multiple agents are dying at the same time, a randomly selected agent will survive. At least one agent runs on a platform.

[3]All agents migrate from a platform whose energy gain is greater than T_B, except a randomly selected agent. If there is only one agent in a platform, the agent cannot migrate. At least one agent runs on a platform.

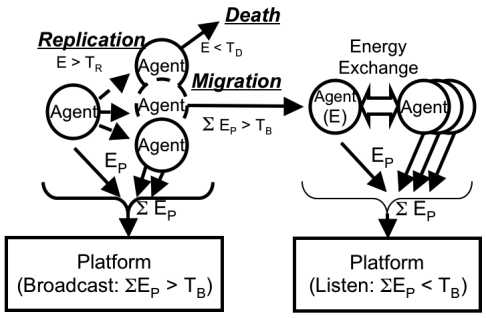

Figure 3. Agent Behaviors

3.2. BiSNET Platform

Each platform consists of two parts: *runtime services* and *state controller*. *Runtime services* hide lower-level computing and networking details (e.g., network I/O), and provide high-level services that agents use to read sensor data and perform behaviors (see also Figure 1).

State controller dynamically changes the state of a sensor node to control its duty cycle. Each sensor node can be in the *listen*, *broadcast* or *sleep* states (Figure 4). A platform and agents can work on a sensor node when its state is in the listen or broadcast state. In either state, each agent performs a series of actions described in Figure 2.

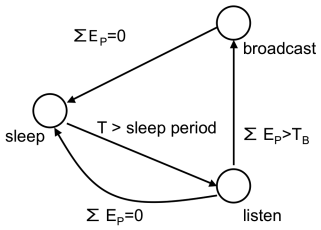

Figure 4. Platform State Transition

In the listen state, a platform turns on a radio receiver to receive data (agents) from neighboring sensor nodes. The listen state changes to the broadcast state if a platform gains energy more than the broadcast threshold ($\sum E_P > T_B$; see also Figures 3 and 4). In the broadcast state, a platform turns on a radio transmitter to allow agents to migrate to neighboring sensor nodes.

When a platform gains no energy from agents ($\sum E_P = 0$), the platform goes into the sleep state (Figure 4). Sleep period is determined as follows. E_{sleep} is a constant.

$$\text{sleep period} = \begin{cases} \frac{E_{\text{sleep}}}{\sum E_P} & \text{if } \sum E_P > 0 \\ E_{\text{sleep}} & \text{if } \sum E_P = 0 \end{cases} \quad (5)$$

The less a platform's energy gain ($\sum E_P$) is, the longer the sleep period is. This means that the platform decreases its duty cycle when agents do not deposit much energy (i.e., when agents find no significant changes in their sensor readings), thereby reducing power consumption. In contrast, when agents find a significant change in their sensor readings, the platform increases its duty cycle in response to higher energy deposit from agents. This allows the agents to collect more data on the environmental change and report them to base stations.

This adaptive duty cycle management mechanism is designed with inference capability in mind. When a sensor node receives agents from neighboring nodes, it decreases its duty cycle even if it has not found any change in its sensor reading. Since the migrating agents bring high energy, the total energy gain of the platform ($\sum E_P$) increases, and this shortens sleep period (see Equation 5). This way, agents can be more watchful for a potential environmental change in the future so that they do not miss it during sleep period.

4. Preliminary Simulation Results

This section shows simulation results to evaluate BiSNET in terms of autonomy, adaptability, self-healing, inference, and power efficiency. BiSNET is implemented on TinyOS and evaluated in the TOSSIM simulator [8].

4.1. Application: Wildfire Detection

This simulation study emulates a sensor network deployed in a forest to detect wildfires. As shown in Figure 5, this simulated network consists of 30 temperature sensors in a grid topology, and a wildfire moves from northeast to southwest. This paper focuses on Nodes 21 and 6. Node 21 detects a temperature change first, and then Node 6 detects it next. Figure 5 shows how the two nodes sense temperature over time.

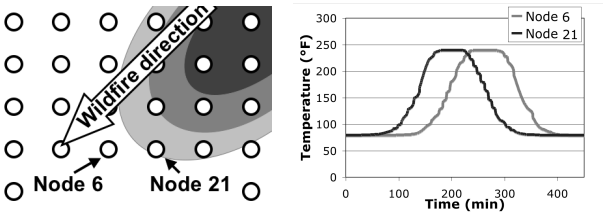

Figure 5. Simulation Configuration

4.2. Adaptive Data Transmission

Figure 6 shows the number of agents migrating from Node 21 to neighboring nodes. Agents migrate to neighboring nodes only when they find rapid changes in their temperature sensing. They do not perform migration at all when there is no temperature changes. BiSNET allows sensor nodes to adapt sensor data transmission to environmental changes (temperature changes).

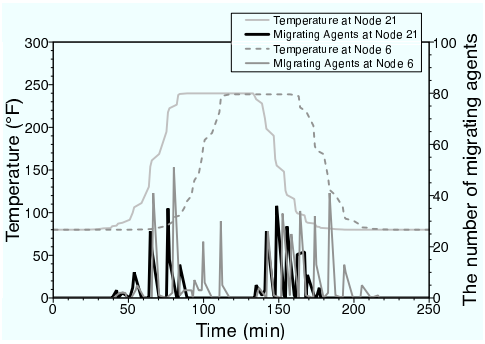

Figure 6. The Number of Migrating Agents

4.3. Inference

Figure 6 shows the number of agents migrating from Nodes 21 and 6. When agents replicate themselves at Node 21 in response to a temperature change, some of the replicated agents migrate to Node 6. The migrating agents convey high energy; the energy is shared with the agents on Node 6, and the agents start replication at Node 6 before temperature changes. This allows agents to be more responsive so that they can immediately migrate toward base stations once temperature changes. In Figure 6, the responsiveness (the time lag between a temperature change and agent migration) of Node 6 is six times better than that of Node 21, thanks to the inference capability in BiSNET.

Figure 7 depicts how sleep period dynamically changes on Nodes 21 and 6. Since migrating agents convey high energy from Node 21 to 6, the platform energy gain ($\sum E_P$) increases at Node 6, which in turn decreases the sleep period of Node 6 (see Equation 5). As shown in Figure 7, sensor nodes (platforms) adapt their duty cycles according to temperature changes. Also, as a result of the inference, Node 6 immediately shortens its sleep period once temperature changes. Its responsiveness (the time lag between a temperature change and duty cycle shortening) is six times better than that of Node 21.

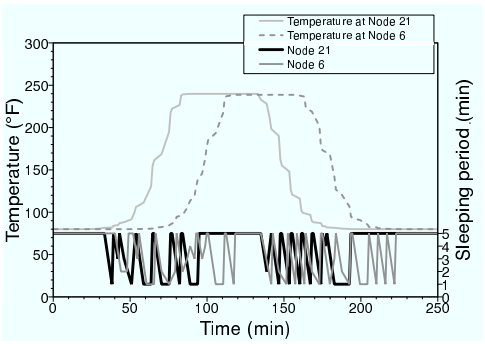

Figure 7. Sleep Period

As described above, the inference capability in BiSNET allows sensor nodes to collect more sensor data to be watchful for potential environmental changes. However, at the same time, they need to consume more power for inference. Table 1 summarizes this tradeoff. It shows the number of collected sensor data and power consumption at the node that performs inference (i.e., Node 6) and the node that does not perform it (i.e., Node 21). The data collection and power consumption are measured between when temperature spikes from 80 to 240 degrees and when it drops back to 80 degrees (for 200 minutes approximately). As shown in Table 1, by drawing inference from the agents migrating from Node 21, Node 6 collects 18.45% more data with only 4.5% more power consumption. The authors of the paper believe that power consumption is small enough to draw inference and BiSNET balances the tradeoff between sensing responsiveness and power consumption.

Table 1. Data Collection and Power Consumption of Node 6

	# of collected data	Power consumption
Without inference (Node 21)	271	2650 mA
With inference (Node 6)	321	2770 mA
Rate of increase	18.45%	4.5%

4.4. Power Efficiency

Figure 8 shows the power consumption of Nodes 21 and 6. In the beginning of a simulation, the sensor nodes consume power to discover neighboring sensor nodes and set up network topology. After that, they minimize power consumption by increasing their duty cycles because there is no significant changes in their sensor readings. When temperature spikes, the power consumption of the sensor nodes spikes too because they immediately decrease their duty cycles (see also Figure 7). As shown in Figure 8, the adaptive duty cycle mechanism in BiSNET allows sensor nodes to effectively save their power consumption when there is no significant environmental change.

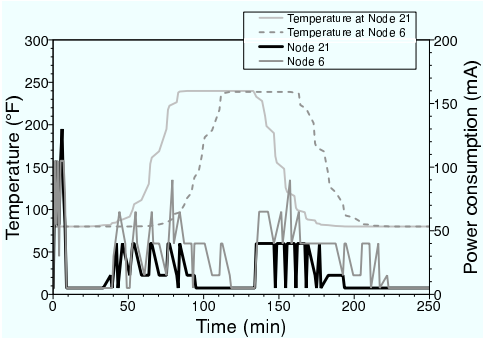

Figure 8. Power Consumption

In BiSNET, sensor nodes dynamically adjust their duty cycles between one and five minutes (see Table 1 and Figure 7). Table 2 compares the power consumption of BiSNET with that of other fixed (one and five minutes) duty cycle mechanisms. Power consumption is measured between when temperature spikes from 80 to 240 degrees and when it drops back to 80 degrees (for approximately 200 minutes). Compared with the five minutes (fixed) duty cycle mechanism, BiSNET consumes only 7% more power. BiSNET sacrifices the 7% power consumption to improve the sensing responsiveness against environmental changes. The five minutes duty cycle mechanism cannot responsively sense and report environmental changes as BiSNET does. Compared with the one minute (fixed) duty cycle mechanism, BiSNET consumes only 20% of power used by one minute duty cycle mechanism. BiSNET effectively reduces power consumption by decreasing duty cycle only when necessary.

Table 2. Power Consumption in Different Duty Cycle Configurations

Duty cycle (sleep period)	Power consumption
1-5 mins (variable; BiSNET)	2650 mA
5 mins (fixed)	2480 mA
1 mins (fixed)	13220 mA

4.5. Self-Healing of False Positive Data

Figure 9 shows a result of self-healing. In this case, Node 21 is configured to malfunction and generate temperature data of 0 and 200 degrees repeatedly. When Node 21 starts malfunctioning, agents perform replication very often and many agents migrate to neighboring nodes because sensor data widely swings between 0 to 200 degrees. (agents' energy intake is very high.) However, the replication thresholds of agents rapidly grow as the agents' energy levels increase (see also Equation 3); within two minutes, agents start suppressing their replications. In five minutes, the number of migrating agents dramatically drops, and eventually, no agents migrate to neighboring nodes even if Node 21 keeps malfunctioning. BiSNET allows sensor nodes to autonomously self-heal false positive data (agents) generated on the nodes and avoid wasting battery power.

5. Related Work

In order to synchronize clocks of sensor nodes in a decentralized manner, [9] applies firefly phase synchronization mechanism in which fireflies synchronize their light on/off periods with each other. BiSNET focuses on different issues; it applies biological mechanisms to adaptive duty cycle management for power efficiency, inference on potential environmental changes for sensing responsiveness and self-healing of false positives in sensor readings.

Agilla proposes a programming language to implement mobile agents for sensor networks, and provides a runtime

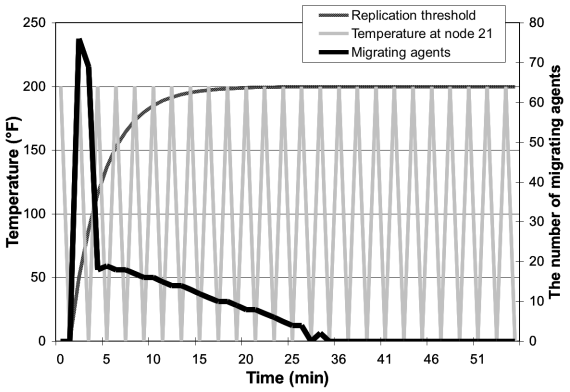

Figure 9. Self-Healing of False Positive Data

system (interpreter) to operate agents on TinyOS [10]. BiSNET does not focus on investigating a new programming language for sensor networks. BiSNET agents and Agilla agents have similar behaviors such as migration and replication. Both of them are also intended to be used in similar applications (e.g., wildfire detection). However, Agilla does not address the research issues that BiSNET focuses on: power efficiency, inference and self-healing.

[11, 12, 13] describe dynamic duty cycle management in sensor nodes. Their goal is to improve power efficiency, and they do not consider sensing responsiveness for potential environmental changes (i.e., the risk to miss significant environmental changes during sleeping period). Unlike them, the duty cycle management scheme in BiSNET is designed to adaptively handle tradeoff between power efficiency and sensing responsiveness for potential environmental changes. As a result, BiSNET uses sensor data to determine the sleep period of sensor nodes, while [11, 12, 13] use other metrics such as the average packet processing time or randomly change sleep period.

Quasar proposes a data collection protocol that handles the tradeoff between data accuracy and power efficiency [14]. In Quasar, a central server controls the state of each sensor node (active or sleep) based on the change in sensor readings. BiSNET does not require any central server; individual sensor nodes locally adjust duty cycle. In addition, BiSNET implements two ways to trigger dynamic duty cycle adjustment: based on the changes in sensor readings and via inference from sensing activities of neighboring sensor nodes. Quasar does not consider inference function.

SASHA proposes a self-healing mechanism by applying immunological mechanisms for base stations to identify fault sensor nodes [15]. A base station detects fault nodes by comparing data from multiple sensor nodes. In BiSNET, individual sensor nodes self-heal false positive sensor data in a decentralized manner. Since false positive data are not transmitted to base stations, BiSNET incurs less power for self-healing.

6. Conclusion

This paper describes a biologically-inspired sensor network architecture, called BiSNET, which addresses several key issues in wireless sensor networks such as autonomy, adaptability and self-healing. Simulation results show that BiSNET allows sensor nodes to autonomously adapt their duty cycles for battery efficiency, to draw inference on potential environmental changes from sensing activities of neighboring nodes, to collectively self-heal (i.e., detect and eliminate) false positives in sensor readings.

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Elsevier Journal of Computer Networks*, vol. 38, pp. 393–422, 2002.

[2] P. Rentala, R. Musunuri, S. Gandham, and U. Sexena, "Survey on sensor networks," in *Proc of Int'l Conf. on Mobile Computing and Networking*, 2001.

[3] G. Mathur, P. Desnoyers, D. Genesan, and P. Shenoy, "Ultra-low power data storage for sensor networks," in *Proc. of IEEE/ACM Conf. on Information Processing in Sensor Networks*, April 2006.

[4] D. Gordon, *Ants At Work: How an Insect Society is Organized*. The Free Press, 1999.

[5] R. Albert, H. Jeong, and A. Barabasi, "Error and attack tolerance of complex networks," *Nature*, vol. 406, July 2000.

[6] N. Minar, K. H. Kramer, and P. Maes, "Cooperating mobile agents for dynamic network routing," *Software Agents for Future Communications Systems*. Springer, 1999.

[7] G. Cabri, L. Leonardi, and F. Zambonelli, "Mobile-agent coordination models for internet applications," *IEEE Computer*, February 2000.

[8] V. Shnayder, M. Hempstead, B. rong Chen, G. Werner-Allen, and M. Welsh, "Simulating the power consumption of large-scale sensor network applications," in *The Second ACM Conf. on Embedded Networked Sensor Systems (SenSys'04)*, November 2004.

[9] G. Werner-Allen, G. Tewari, A. Patel, M. Welsh, and R. Nagpal, "Firefly-inspired sensor network synchronicity with realistic radio effects," in *Proc. of The Third Int'l Conf. on Embedded Networked Sensor Systems*, 2005.

[10] C.-L. Fok, G.-C. Roman, and C. Lu, "Rapid development and flexible deployment of adaptive wireless sensor network applications," in *Proc. of Int'l Conf. on Distributed Computing Systems*, June 2005.

[11] C.-F. Hsin and M. Liu, "Network coverage using low duty-cycled sensors: Random & coordinated sleep algorithms," in *Proc. of The Third Int'l Symposium on Information Processing in Sensor Networks*, 2004.

[12] J. Mišić and V. B. Mišić, "Duty cycle management in sensor networks based on 802.15.4 beacon enabled MAC," *Ad Hoc and Sensor Wireless Networks Journal*, vol. 1, no. 3, pp. 207–233. Old City Publishing, Inc., 2005.

[13] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM trans. on Networking*, vol. 12, no. 3, pp. 493–506, 2004.

[14] Q. Han, S. Mehrotra, and N. Venkatasubramanian, "Energy efficient data collection in distributed sensor environments," in *Proc. of The Twenty Fourth IEEE Int'l Conf. on Distributed Computing Systems*, March 2004.

[15] T. Bokareva, N. Bulusu, and S. Jha, "SASHA: Towards a self-healing hybrid sensor network architecture," in *Proc. of The Second IEEE Int'l Workshop on Embedded Networked Sensors*, May 2005.