A Cognitive Approach for Gateway Relocation in Wireless Sensor Networks

Waleed Youssef and Mohamed Younis
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, MD 21250
Email: youssef1@umbc.edu, younis@umbc.edu

Abstract

Wireless sensor networks are composed of a number of sensors probing their surroundings and disseminating the collected data to a gateway node for processing. Numerous military and civil applications have emerged over the last few years for wireless sensor networks. In many of these applications, sensors and gateways are placed in harsh environments. Therefore, protecting snsors and the gateway is critical for ensuring the robustness of the network operation. Gateway relocation has been pursued as means for boosting network-related performance metrics, such as throughput and energy consumption. However, we argue that relocating without taking safety concerns into consideration may cause the gateway to move dangerously close to one or multiple serious events in the environment. In this paper, we present GRASP, a new algorithm for Gateway Relocation for Adaptive Safety and Performance requirements/goals. GRASP employs an evolutionary neural network to estimate the risk at the new proposed gateway position before the relocation process takes place. Our experimental validation results have demonstrated the effectiveness of GRASP.

Introduction and Preliminaries

In the past few years, *wireless sensor networks* (WSN) have received increasing interest from the scientific and engineering communities due to their potential use in many applications such as target tracking, disaster management, border control and battlefield surveillance (Akyildiz *et al.* 2002; Estrin *et al.* 1999; Karl & Willig 2005; Min *et al.* 2001; Pottie & Kaiser 2004). Components of the WSN include sensors, gateways, command nodes, and monitored phenomena (Figure 1). Sensors are miniaturized battery-operated devices equipped with a communication subsystem. The main function of WSN is to collect data and report any abnormal conditions happening in an area of interest. These conditions are reported by close-by sensors and transmitted as packets, often over multi hop paths, to a central unit called gateway for analysis. The gateway interfaces the WSN to command and control centers, reporting serious findings and/or required data. The reported data may indicate that some moving targets crossed in from outside the deployment area, the irruption of a fire, etc. Sensors consume energy every time they are involved in data transmission. Thus, highly active sensors tend to lose their energy faster than inactive ones. Once all the onboard energy supply drains, the sensor becomes dysfunctional. Therefore, an important goal in WSN is to minimize the transmission power in order to keep sensors alive for the longest time possible.

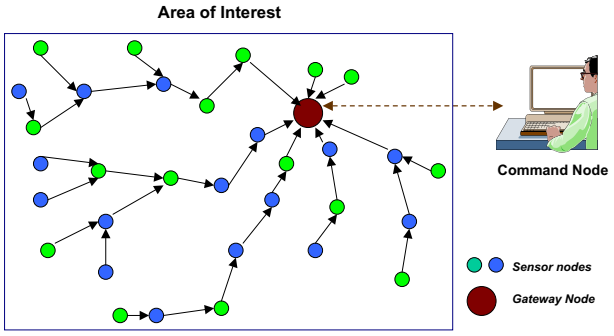


Figure 1: A sample sensor network where packets are transferred through sensors to reach the gateway and then to the command center

Gateway relocation is one of the approaches pursued to improve the performance of the network (Akkaya *et al.* 2005; Tirta *et al.* 2004). By relocating the gateway towards highly active sensors, the packets will be routed to the gateway through fewer sensors. The shortened data paths help in preserving sensors' energy, lowering the packet loss rate and reducing delivery latency. In general, most gateway relocation techniques identify bottlenecks in the current network topology and tend to move the gateway close to these bottleneck positions. However, such performance-centric relocation may move the gateway dangerously close to one or multiple targets/events in the environment and thus may expose the gateway to the risk of getting damaged, captured, etc. In this paper, we present a solution to the gateway relocation problem that balances between the WSN performance and the gateway safety goals. The idea is to identify a position for the gateway to better serve the area/event that triggers the most data traffic without negatively impacting the tracking of other events and at the same time not put the gateway

at risk. Since the physical security of the gateway is usually measured qualitatively, we employ decision support systems to generate a quantitative assessment.

A *Decision Support System* (DSS) is an intelligent module that makes decisions based on historical data. The process for enabling such an intelligent decision-making system consists of three main steps. First step is to collect historical data and use it to train the DSS. Second step is to define and build the decision model. Third step is to utilize the trained DSS in making future predications and decisions. Recent developments in DSS have shown that heuristics - including artificial neural networks, fuzzy sets, and genetic algorithms - offers opportunities for combining multiple criteria and exploring new patterns, which eventually enhances the quality of the decision making process (Anderson 1995; Chen *et al.* 2004; Negnevitsky 2002). The selection of the right heuristics to use is problem-dependent and varies based on many different factors (Negnevitsky 2002). *Artificial Neural Networks* (ANN) proved to be superior to traditional statistical methods when data exhibit unpredictable non-linearity, when patterns important to the decision-making process cannot be clearly identified, or when data are fuzzy in nature, involving human opinions or subject to some degree of uncertainty.

Many related work have studied the effect of gateway relocation on WSN performance in terms of energy, throughput and latency. The reported results in (Akkaya *et al.* 2005; Akkaya & Younis 2004; Younis *et al.* 2003) showed that a considerable amount of improvement is achieved by relocating the gateway to a location where traffic volume is the highest within the event area. Such repositioning of the gateway increases the average lifetime of the sensor nodes by decreasing the average energy consumed per packet. To achieve the same goal, continual gateway mobility is also pursued in (Akkaya & Younis 2004; Kim *et al.* 2003; Luo *et al.* 2005). However, none of these approaches takes into account any safety concerns related to the movement of the gateway node. In most environments where WSN are employed, the movement of the gateway can be restricted by existing risks such as enemy soldiers, tanks, obstructions and fires etc. In those cases, the gateway should also consider the risk level for its movement so that it will not be damaged or captured. To the best of our knowledge, no similar work has been done so far for handling such a problem in WSN. Our work will be among the first to provide performance improvements for the sensor network in addition to caring for the gateway's safety.

The organization of this paper is as follows. The next section covers the artificial neural networks model used. Section describes our approach and discusses few implementation issues. Validation results are presented in Section . Finally, Section concludes the paper and outlines our future research plan.

The Artificial Neural Network Model

The general structure of an artificial Neural Network (ANN) consists of many neurons that are interconnected together to form one or more layers. Each neuron can be considered as a simple processor with collective behavior, where its inputs are weighted variables and its output is a function of the inputs, their associated weights, and the threshold value. One of the main features of neural networks is their distributed associative memory property, in which the information is stored in the weighted links, rather than at specified memory addresses. An ANN can be considered as a generic learning machine that has been proven to be capable of forming representations of complex and non-linear phenomena (Anderson 1995; Negnevitsky 2002).

The topology of an ANN is problem-dependent but usually is a multi-layer network with either forward feed or backward propagation links. In forward feed networks, the input of one layer is the output of the previous layer with no links going backwards. However, in recurrent neural networks, the output of one layer can be fed to the input of a previous layer (Negnevitsky 2002). For an ANN to provide the desired output, it has to go through a learning phase first. The learning phase is a multi-step process usually performed after identifying the topology of the ANN. Learning methods include back propagation or evolutionary techniques. Evolution of neural networks using evolutionary algorithms has gained popularity in recent years giving rise to a new branch known as Evolutionary Neural Networks (ENN) (Yao 1999).

In this paper, we pursue heuristics to train the neural network and find values for all weights and thresholds. We employ an ENN to assess the risk at new locations before moving the gateway. The objective of the ENN is to find a better and safer location in the vicinity of the proposed gateway's position that boosts (does not degrade) the performance. For an $n-$input ENN, the output is a risk factor r that indicates the threat that the gateway will be exposed to at the new location. Each input to the ENN represents the distance d_i between current gateway location and event # i in the environment. The general formula for r is:

$$r = \sum_{i=n^2+1}^{n^2+n} \left[w_i \left(\sum_{j=1}^{n} d_{jl} w_{[(i-n^2)+(j-1)*n]} \right) - T_{i-n^2} \right] - T_{n+1}$$

where w_i are the weights, and T_j are the threshold values.

Since the historical data used to train the model consists of multiple positions, the risk factor output would be a vector of size equal to the cardinality of the historical data set. In the next section, we describe GRASP in details.

Safe Gateway Relocation Algorithm

In this paper, we present GRASP, a new algorithm for Gateway Relocation for Adaptive Safety and Performance using evolutionary neural networks. GRASP is different from other methods, such as the one described in (Akkaya *et al.* 2005), that only emphasize improvements in performance (throughput and energy consumption) of the WSN. In (Akkaya *et al.* 2005), the geometric centroid of all sensors with high-energy consumption is identified, and the gateway is relocated to this point in order to reduce energy expenditure by surrounding sensors. This relocation method works fine when there are no dangerous events/targets in the environment. In other situations where harsh conditions and

safety threats are often a problem, relocating the gateway based on performance measures only may cause the gateway to malfunction. Adding a security expert component to the repositioning process would improve both the performance of the network and the safety of the gateway.

GRASP consists of three modules that perform the following functions: collect historical data, use genetic algorithms to resolve ANN weights and thresholds, and finally utilize the resultant ENN to make relocation decisions and predications. Details about these modules are provided in the next few subsections.

The GRASP-History Module

In this module, the past gateway locations are manipulated to generate the learning data set. The manipulation process starts by estimating the threat level at each location the gateway visited. The threat level is estimated as a function of the number of events in the environment, the proximity of the gateway to the target, and the severity of the events. Since the number of locations the gateway can relocate to is infinite, a method was developed to discretize the environment and evaluate the risk probability at each discrete location. The idea is to divide the deployment area into a two dimensional grid of size $m \times m$, resulting in m^2 cells. The risk probability at the center of each cell location is calculated as:

$$risk(cell(l)) = t_{high}/(t_{high} + t_{low})$$

where t_{high} represents number of times the next gateway location was a higher threat level point than the current location, and t_{low} represents number of times the next gateway location was a lower threat level point. Figure 2 outlines the GRASP-History module.

```
Algorithm GRASP-History (HstryDataSet)
    For each location l ∈ HstryDataSet
        T_l ← estimate threat level(l)
    cell[i, j] ← cluster Area into m × m
                 2-dim grid;
    For each cell location
        t_h ← Count(gateway moves to locations
                    with higher threat level);
        t_l ← Count(gateway moves to locations
                    with lower threat level);
    For each location l ∈ cell[i, j]
        Pr{Risk_l} = t_h/(t_h + t_l)
End;
```

Figure 2: The GRASP-History module

The GRASP-GA Module

In this module, we use genetic algorithms to efficiently and effectively identify all the weights and thresholds of the ENN (Goldberg 1989). Each learning data sample would yield an equation in $(n^2 + 2n + 1)$ variables, which represents the total number of weights and thresholds in the ENN. The number of equations equals the cardinality of the learning data set provided. The GRASP-GA module is outlined in Figure 3. Details about the implementation of each component of the genetic algorithm are provided next.

```
Algorithm GRASP-GA ( LD: Learning data)
    generate a random initial population P
    repeat
        Select two parents p1 and p2
        u ← crossover(p1, p2)
        LocalOptimize(u, LD)
        mutate(u)
        replace(u, p1, p2, P, LD)
    until (there is no improvement)
    return the best member of P;
End;
```

Figure 3: The GRASP-GA module

- **Encoding.** Each individual member in the population is represented as an array of floating point coefficients. Each entry in the array corresponds to a single coefficient in the neural network. The size of the array is dependent on the number of events in the environment. For example, if n is the number of events reported, then the individual member length is $n^2 + 2n + 1$.
- **Initial Population.** An initial population of size 100 is randomly created. The size of the population remains constant throughout the algorithm. Also, all members of the population have the same individual length.
- **Fitness.** The fitness of each member in the population is calculated as the total hamming distance between the individual and each member in the learning data set. The less the total hamming distance, the more fitted the individual member is.
- **Parent Selection Scheme.** We used the Tournament selection scheme in GRASP. This method adopts a tournament-like competition, where four parents compete with each other in a pair-wise competition. The best two parents are selected and returned while the worst member is saved for the Replacement scheme.
- **Crossover Operator.** We used a fixed 3-point crossover operator in GRASP. The fitness of the two resulting offspring are calculated and the one with the better fitness is returned for mutation.
- **Mutation Operator.** Mutation is used mainly to introduce some randomness into the population. In GRASP, we used a 10% mutation operator, where mutation is performed on each newly created offspring as follows. For each chromosome of the offspring and with probability p, the chromosome is selected for mutation. Its value is then changed by a randomly generated amount δ and the new value is stored.
- **The Local Optimization Process.** After mutation, the new offspring is optimized. The local optimization process has two main stages. The first stage is to scan the input mutated offspring, and for each chromosome of

the offspring, slightly change its value and check the effect of this change on fitness. If fitness improves, keep the change; otherwise discard it. The second stage is to change the threshold value. This is done by summing up the input values, and then the threshold is set to the negative of this value. The fitness of the new sequence is calculated and new optimized sequence is stored if its fitness is better than the fitness of the original sequence.

- **Replacement Scheme.** The optimized offspring is compared with the worst member obtained from the tournament selection method. If the fitness of the offspring is better, the member is replaced in the population. Otherwise, the offspring is discarded.
- **Stopping Criteria.** The genetic algorithm stops after reaching certain number of iterations ($50,000$ in the experiments) or after evolving for many iterations ($1,000$) with no improvements in fitness of the whole population.

The GRASP-Location Module

The GRASP-Location module is responsible for making the safety/performance decision. It utilizes the resultant ENN to relocate the gateway to a safer location than the current location. The ENN handles the safety estimation of the gateway, however, for the WSN performance, GRASP-Location uses the network throughput and the average remaining energy of some relaying sensors in its vicinity to make the performance decision. A predefined importance weights is set at the beginning of the module to indicate the relative importance of the performance and the security in the relocation function, i.e. the objective function. The objective function returns an index, called the *Relocation Index* for each input location. GRASP-Location implements a depth-constrained algorithm that exhaustively searches the vicinity of the proposed gateway location and within a predefined radius r. The algorithm divides the area into coronas and wedges and the location that has the best value of the objective function is selected. As an example, the relocation index (RI) at point l_1 can be calculated as:

$$RI(l_1) = (a * SafeIndex(l_1)) + (b * PerfIndex(l_1))$$

{where a is the risk weight, and b is the performance weight.}

Experimental Validation

In this section, we describe the validation of GRASP in a simulated target tracking application setup similar to that used in (Akkaya *et al.* 2005). The simulator was implemented, along with all algorithms for GRASP, using C++ and was run on a PC with Pentium IV 2.4GHz Intel processor with 512MB of RAM. The organization of this section is as follows. We start by describing the experimental setup. We follow by briefly discussing conducted experiments then we give details about selected sample experiments and results obtained. We then summarize all results obtained. Finally, we present results and statistics about the complexity of the genetic algorithm.

Setup and Metrics

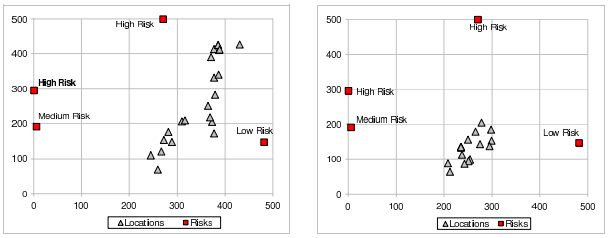
In all the experiments, the network consists of varying number of sensor nodes (50 to 200) that are randomly placed in a $500 \times 500\ m^2$ area. Each node is assumed to have an initial energy of 5 joules. For each conducted experiment, we collected safety and performance results. For safety, the distance between the gateway and each target was reported. For performance, we used the network throughput and the average energy per packet. In all experiments, three approaches were compared (1) when relocation was not allowed (2) when the gateway was allowed to relocate based only on performance metrics only (3) when using GRASP. In the genetic algorithm, Mersenne Twister random number generator was used to generate uniformly distributed random numbers (Matsumoto & Nishimura 1998).

Experimental Results

For the first and second approach, we conducted 9 experiments each. Meanwhile, for GRASP, we conducted the following experiments. We tested 5 different numbers of targets in the environment, (3, 4, 5, 6, and 8). We studied the impact of changing the grid size when generating the learning data, as in the set (50, 100, 125, 250). For risk/performance weights of the objective function in the GRASP-Location module, we tested 5 different setups, ((0, 1), (0.2, 0.8), (0.5, 0.5), (0.8, 0.2), and (1, 0)). For the search radius r of the GRASP-Location algorithm, we used 4 different radius, as in the set (10, 20, 30, and 50). Thus, the total number of experiments performed was 18. Moreover, we have applied 5 distinct seeds in order to generate random network topologies. Each experiment lasted $12,000$ sec. We observed that with confidence level $> 90\%$, the simulation results stayed within $6\% - 10\%$ of the sample mean. Due to space limitations, we report on the results of a subset of the conducted experiments.

Experiment 1 In this experiment, we studied the effect of using GRASP on positions visited by the gateway. We have conducted two tests. First, we applied the performance-based relocation scheme of (Akkaya *et al.* 2005). Figure 4(a) shows a plot of some locations visited by the gateway during this sample experiment. The second test involved GRASP. Figure 4(b) shows locations visited by the gateway in this case. It is clear from the first figure that the gateway jeopardizes its safety by moving dangerously close to one or more targets in the environment. While, through GRASP, the gateway managed to stay further away from risky targets to ensure its safety.

Experiment 2 In this experiment, the effect of increasing the number of targets on the WSN was examined. The number of targets was set to (3, 4, 5, 6, or 8). Figure 5 shows that the gateway managed to securely relocate to positions that maintained safe distance from targets and provided acceptable network performance. With the increase in number of targets, GRASP proved to be an effective technique and continued to appropriately select the right position for the gateway. The performance plots in Figure 6 shows that the simulation results were comparable to these results for the Performance-Relocation technique and were better than the No-Relocation technique. The results of this experiment confirmed the effectiveness of the GRASP approach.



(a) Performance-Reloc results (b) GRASP results

Figure 4: [a] A sample experiment showing locations visited by the gateway using the Performance-Relocation approach when there were 4 targets reported. [b] Gateway locations visited for the GRASP approach

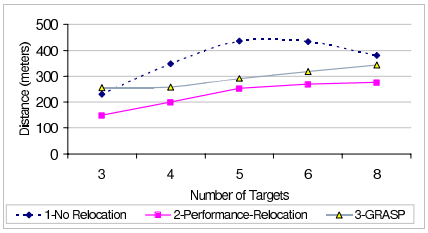


Figure 5: Average distance between the gateway and targets for all approaches tested when changing number of targets in the environment

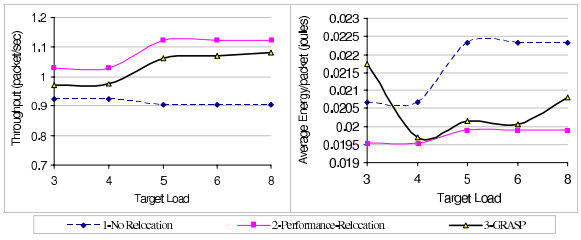


Figure 6: Effect of changing number of targets on WSN performance for experiment set #2

Experiment 3 In this experiment, we studied the effect of changing the cell size when generating the learning data on the gateway safety and performance of the WSN. We used values in the set (50, 100, 125, and 250). For example, setting the grid size to 50 means that the deployment area was divided into a grid of size 10×10, resulting in learning set with cardinality of 100. Figure 7 shows that the average distance between the gateway and targets has increased after using GRASP for fine grained grid sizes, implying an increased gateway safety. In addition, Figure 8 shows that the network performance was improved over the No-Relocation technique for different cell sizes. The degradation in performance compared to the Performance-Relocation technique was minimal. The fluctuation in the curve has resulted from two things (1) the randomness in the genetic algorithm (2) the quality of the historical data collected for each experiment. It is clear that for all grid size, GRASP was able to

Table 1: Summary of both network performance and gateway security results for all performed experiments

Process	Security	Network Performance	
		T-put	Energy
Perf-Reloc (compared to No-Reloc)	-41.82%	19.28%	-8.57%
GRASP (compared to No-Reloc)	-32.50%	11.01%	-5.14%
GRASP (compared to Perf-Reloc)	16.03%	-7.46%	-3.62%

enhance the security of the gateway and at the same time some improvements in network performance were noticed. The choice of which grid size to use is dependable upon the cardinality of the historical data.

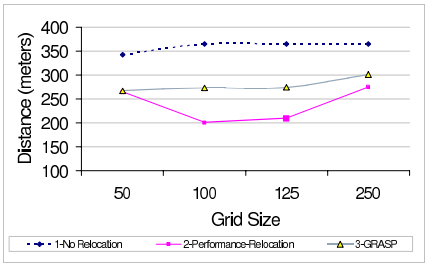


Figure 7: Average distance between the gateway and targets for all approaches tested when changing the grid size

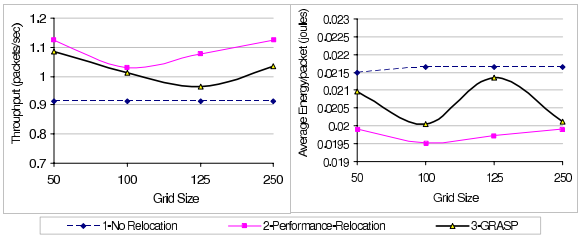


Figure 8: Effect of changing the grid size (the resolution of the deployment area when generating the learning data) on the WSN performance when GRASP is enabled for experiment set #3

Summary Results

Table 1 summarizes all results obtained for all approaches tested relative to each other. It is observed that the GRASP approach was able to enhance the safety of the gateway compared to the Performance-Relocation approach. In addition, GRASP was able to enhance and boost the network performance compared to the No-Relocation approach. GRASP was able to balance the safety and performance goals of the gateway. All these results have confirmed the effectiveness of GRASP in securing the gateway while boosting the network performance.

Genetic Algorithm Complexity

In GRASP, we employed genetic algorithms to train neural networks. Such approach has been in use for a while. However, a concern about the time complexity of the algorithm always arises. Figure 9 illustrates some results obtained from the genetic algorithm. It shows the effect of changing number of targets as well as the grid size on the running time and quality of the solution obtained. The figure also confirms that the complexity and quality of solutions in GRASP stayed always within the acceptable range and did not burden the system with more than anticipated computational overhead.

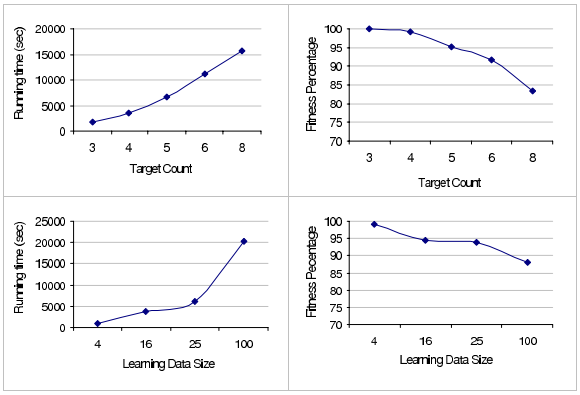


Figure 9: Effect of changing the number of targets and the grid size on GRASP-GA results

Conclusion

In wireless sensor networks, data are usually routed from sensors to a gateway node, where it is further processed. Since the position of the gateway has a significant impact on the network performance, changing the location of the gateway is sometimes pursued to optimize the throughput, delay and energy consumption. However, such performance centric process may reposition the gateway in the proximity of harsh events or dangerous targets and thus may get it exposed to increased risk. To tackle this issue, this paper introduced GRASP, a new adaptive safety and performance aware algorithm for gateway relocation in wireless sensor networks. The main goal of GRASP is to balance between protecting the gateway and the desire for enhancing the network performance. GRASP employs evolutionary neural networks to assess the risk involved in placing the gateway at various positions in an area of interest and identifies locations that gateway can move to. Experimental results confirmed the effectiveness of GRASP. In the future, we plan to extend GRASP to use other performance metrics like timeliness. We also plan to include the cost of the relocation as one of the factors affecting the process.

References

Akkaya, K.; Younis, M.; and Bangad, M. 2005 "Sink Repositioning for Enhanced Performance in Wireless Sensor Networks," *Computer Networks*, 49: 512-434.

Akkaya, K. and Younis, M. 2004. "Relocation of gateway for enhanced timeliness in wireless sensor networks," *Proc of IEEE Workshop on Energy-Efficient Wireless Communications and Networks (EWCN)*, Phoenix, AZ.

Akkaya, K., and Younis, M. 2004. "Energy-Aware Routing to a Mobile Gateway in Wireless Sensor Networks," *Proc of the IEEE Globecom Wireless Ad Hoc and Sensor Networks Workshop*, Dallas, TX.

Akyildiz, I. F. et al., 2002. "Wireless sensor networks: a survey", *Computer Networks*, 38: 393-422.

Anderson, J. A., 1995. *An Introduction to Neural Networks*, MIT Press.

Chen K.; Jacobsom, C.; Blong, R. 2004. "Artificial Neural Networks for Risk Decision Support in Natural Hazards: A Case Study of Assessing the Probability of House Survival from Bushfires, Environmental Modeling and Assessment," 9(3): 189-199.

Estrin, D. et al., 1999. "Next Century Challenges: Scalable Coordination in Sensor Networks," *Proc of the Fifth Annual International Conference on Mobile Computing and Networks (MobiCOM '99)*, Seattle, WA.

Goldberg, D. E., 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Press.

Karl H. and Willig, A. 2005. "Protocols and Architectures for Wireless sensor networks," Wiley, United Kingdom.

Kim, H.; Abdelzaher, T.; and Kwon, W. 2003. "Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks," *Proc of the 1st international conference on Embedded networked sensor systems*, Los Angeles, CA.

Luo, H. et al. 2005, "TTDD: Two-tier Data Dissemination in Large-scale Wireless Sensor Networks," *in ACM/Kluwer Mobile Networks and Applications (MONET)*, 11(1-2): 161-175.

Matsumoto, M. and Nishimura, T. 1998. "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator," *ACM Transactions on Modeling and Computer Simulation*, 8(1), 3-30.

Min, R. et al., 2001. "Low Power Wireless Sensor Networks," *Proc of International Conference on VLSI Design*, Bangalore, India.

Negnevitsky, M. 2002 *Artificial Intelligence A Guide to Intelligent Systems*, Addison Wesley, Harlow, England.

Pottie, G. J. and Kaiser, W. J. 2000. "Wireless integrated network sensors," *Communications of the ACM*, 43(5): 51-58.

Tirta, Y.; Li, Z.; Lu, Y-H; and Bagchi, S. 2004. "Efficient Collection of Sensor Data in Remote Fields Using Mobile Collectors," *Proc of the International Conference on Computer Communications and Networks (ICCCN 2004)*, Chicago, IL.

Yao, X., 1999. "Evolving Artificial Neural Networks," *Proc of the IEEE* 87(9): 1423-1447.

Younis, M.; Bangad M. ; and Akkaya, K. 2003. "Base-Station Repositioning For Optimized Performance of Sensor Networks," *Proc of the IEEE VTC 2003 - Wireless Ad hoc, Sensor, and Wearable Networks*, Orlando, FL.