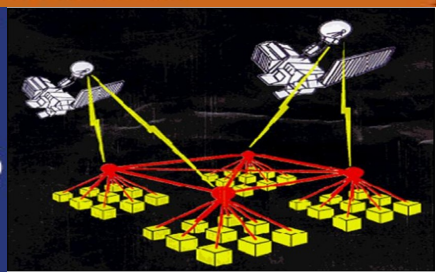
The Third International

Innovations and Real-time Applications of Distributed Sensor Networks (DSN) Symposium

http://www.ira-dsn.org/ November 26-27, 2007 . Shreveport, Louisiana



EDITORS

V. Phoha

S.J. Park

S.S. Iyengar

Shreveport Convention Center
Shreveport, Louisiana

GENERAL CHAIR

Les Guice

Conference Chairs

S. S. Iyengar
Vir V. Phoha

Publication & Proceedings Chair

Seung-Jong Park

Publicity Chair for Announcement and Web

Guna Seetharaman

Local Arrangement Chair

Kristen Martin

TECHNICAL PROGRAM COMMITTEE

S. Vason	**P. Varshney**
N. Rao	**R. Fehling**
R. Brooks	**T. Laurens**
S. Park	**R. Sastry**
G. Allen	**G. Seetharaman**
T. Kosar	**J. Monde**
J. Tsai	**K. Mori**
P. Chen	**D. Kim**

MESSAGE FROM THE GENERAL CHAIR

It is my privilege to welcome to Shreveport all the distinguished guests and participants in the two conferences: The Third International Innovations and Real-time Applications of Distributed Sensor Networks (IRA-DSN) Symposium and the first Cyberspace Research Workshop (CRW).

This is a time of rapid progress in cyberspace technology. If we are to expect significant advances in cyber-centric sensor systems research, we must bring together the best minds from the community of cyberspace, sensor networks, and allied areas. This is the purpose of the joint sessions of IRA-DSN and the CRW. It is the first time that the two conferences have joined together to present a venue to discuss emerging technologies, share ideas, and create research and development opportunities for researchers and practitioners in areas of cyber security and sensor networks.

We are most pleased to be able to host these two research events in conjunction with the 1st Annual Air Force Cyberspace Symposium, being jointly sponsored by the provisional Air Force Cyber Command (AFCyber) and the 8th Air Force, both headquartered at Barksdale Air Force Base. Co-sponsoring the Symposium is the Cyber Innovation Center (CIC), a public-private partnership that is intended to foster collaboration, research, and technology development in the cyberspace industry. This is the first major event to bring together the distinguished government, military, and private sector leaders who have been responsible for establishing the groundwork for AFCyber. We believe that it is critically important for the leading researchers in academia and industry to maintain association with AFCyber and the CIC and be prepared to contribute to their research, education, and innovation missions as these organizations move into full operation.

The State of Louisiana has recently demonstrated its confidence in Louisiana's leading cyber researchers by funding the new Louisiana Tech-LSU Center for Secure Cyberspace (CSC) to establish a tower of research excellence in cyber-centric sensor systems. The Center has initiated research in new areas of interest to both the military and the civil sector. The scope of the papers reflects the focus of the Center's research. We are pleased with the diversity of the topics and broader representation from academic researchers and industry, representing mature work and research in progress.

Perhaps the most important measure of the success of a research conference is the quality of original research and the discussions and ideas that are generated. We hope the joint sessions of IRA-DSN and CRW provide you with a sense of intellectual fulfillment. Most importantly, we hope that you get opportunities to meet other researchers and take back to your institutions many ideas and friendships that will seed new research and collaborations.

Les Guice, General Chair

MESSAGE FROM THE PROGRAM CHAIRS

"*Present*" is the beginning of the confluence of two powerful technologies, the technology underlying the cyberspace and the technology underlying sensor networks. The research spanning both of these technologies has begun to influence the design and development of systems in the civil and the military sector. However, many ideas are still unproven. The time is ripe to provide a venue where researchers can share their ideas in a friendly environment with a lot of constructive criticism from educated peers. The Third International Innovations and Real-time Applications of Distributed Sensor Networks Symposium combined with the first Cyberspace Research Workshop provide such a venue.

Combination of cyberspace and sensor networks provides a broad field and many opportunities for fundamental and applied research. However, to provide a focused coverage, for this year we narrowed the theme of the workshop to secure and optimal placement of soft and hard sensors in unknown and uncertain environments. Our premise is that the sensed information has to be transmitted and processed within the cyberspace, thus providing constraints on the placement algorithms. Each paper represents a unique problem, or a unique solution, or a unique argument, or a unique implementation, or a combination of these.

Three events influenced the choice of Shreveport as the conference venue: (1) The State of Louisiana's funding for a major Louisiana Tech-Louisiana State University research center, the *Center for Secure Cyberspace*, located in Louisiana Tech University just 70 miles east of Shreveport. The Center's activities influenced the scope of these two workshops. (2) The Air Force's major Cyber initiative and the location of a provisional Air Force Cyber Command in Shreveport provides close proximity to a prospective and ready consumer of research in this field. And, (3) the increased investments by the state in creating cyber infrastructure such as Louisiana Optical Network Initiative, and the resultant spurt in activities in cyberspace.

This conference has required the efforts of many people: the keynote speakers, the authors of the papers, the reviewers, the program committee, and others, too numerous to list here. We thank them all.

Vir V. Phoha and S. S. Iyengar, Program Chairs

Proceedings for the 3rd International innovations and Real-Time Applications of Distributed Sensor Networks symposium

PROGRAM – 26th Nov, 2007 (IRA-DSN)

07:30 – 08:25 AM	Registration
08:30 – 08:40 AM	Welcome by Dr. S.S. Iyengar and Dr. Phoha
08:40 – 09:30 AM	Keynote Talk #1: (Modern Traffic Analysis and its Capabilities by Ricardo Bettati, Texas A&M)
09:30 – 10:00 AM	Sensor Stream Analysis for Behavior Recognition - Brooks
10:00 – 10:30 AM	Program Analysis for Sensing Mutant Cyberweapons
10:30 – 10:45 AM	Coffee break
10:45 – 11:15 AM	Robust Clustering Algorithm for Target Tracking in Wireless Acoustic Sensor Networks - Alvin Lim
11:15 – 11:45 AM	On Energy-Efficient Priority-Based Routing in Heterogeneous Sensor Networks - Qishi Wu
11:45 – 12:15 AM	Contour Guided Dissemination in 4-Neighbor Wireless Mesh Topologies - Shivkumar Sastry
12:15 – 01:30 PM	Lunch (Peter Chen – Luncheon Talk – Terrorist Profiling)
01:30 – 02:15 PM	Prof. Sahni (Algorithmic Sensor Networks)
02:15 – 02:45 PM	Sensor Graph Mining - Ali Hurson, Vijay Kumar, Praveen Rao
02:45 – 03:15 PM	Coffee Break
03:15 – 03:45 PM	Role of IBM's SSME discipline in Cyberspace Operations (Ramesh Reddi)
03:45 – 04:15 PM	Estimation of Data Redundancy in Wireless Sensor Networks
04:15 – 04:45 PM	Stream Hierarchy Data Mining - Maggie and Vijay Kumar
04:45 – 05:15 PM	Innovations at Air Force Labs - AFIT
05:15 – 06:00 PM	(Open Slot)
06:00 – 06:15 PM	Reception
07:00 PM	Banquet Talk by Dr. Ram Sastry of Siemens

Title: Modern Traffic Analysis and its Capabilities.

Speaker: Riccardo Bettati, TAMU

One day in the not-too-distant future all confidential communication will be safely encrypted by secure codes and securely exchanged by protocols that work and by users who know what they are doing.

Does this mean that it will be time for agents in surveillance and forensics to call it a day?

When traffic is encrypted, and its contents therefore beyond the reach of effective cryptanalysis, we enter the realm of "traffic analysis". In this talk, I will give an overview of the capabilities of modern traffic analysis, and describe how it can be used to break encryption, to render anonymity systems ineffective, to identify honeypots or bots in multiplayer games, or to help discover the configuration of protected remote systems. I will also show interesting applications of signal processing and sensor networks methodologies to address hard problems in wired and wireless networks. We will also speculate about possibilities of modern traffic analysis in VOIP.

Countermeasures against traffic analysis exist. I will describe a few, and illustrate some unexpected weaknesses of such systems.

Optimal Distributed Denial of Service (DDoS) Attacks

Chinar Dingankar, Sampada Karandikar, and R. R. Brooks,
Holcombe Department of Electrical and Computer Engineering
Clemson University
PO Box 340915
Clemson, SC 29634-0915
Email: rrb@acm.org

***Abstract*—Computer networks are subject to Distributed denial of Service (DDoS) attacks, where a single attacker instructs multiple subverted hosts (zombies) to consume the victim's network resources. The idea presented here extends our initial work. We quantify the security of an enterprise network against denial of service (DoS) attacks by finding the minimum number of zombie processors required to disable the network. We phrase the DDoS problem as a board game played on the network infrastructure and use max-flow min-cut analysis to identify system bottlenecks. Our results provide a reasonable metric of the network resiliency against DDoS, since it quantifies the resources needed by the attacker to stage a successful attack. This information can be used in turn to make the networks less susceptible to attack.**

***Index Terms*—Denial of Service Attacks, Graph Theory, Network Security.**

I. Introduction

A Denial of Service (DoS) attack is an explicit attempt by attackers to prevent legitimate users of a service from using that service [1]. In Distributed DoS (DDoS) attacks a single attacker (*master*) coordinates many hijacked systems (*zombies*) (see Figure 1). They are difficult to trace and it is almost impossible to find the source of a DDoS attack. This makes system recovery a slow and costly process.

A DDoS attack is a networked DoS where nodes work together [2]. A zombie is a daemon that performs the actual attack [3]. On receiving an attack command from a hidden master processor the zombies, located on different computers, launch an attack on the target. Hundreds or thousands of zombies working together swamp the bandwidth of the target's network connections. In the smurf attack, shown in Figure 1, zombies add an extra level of indirection by sending ping packets requesting from random nodes that a response be sent to the victim. The owners of zombie computers are usually unaware their machines are participating in a DDoS attack. This, and the indirect nature of the attack shown, makes it almost impossible to find the source of an attack.

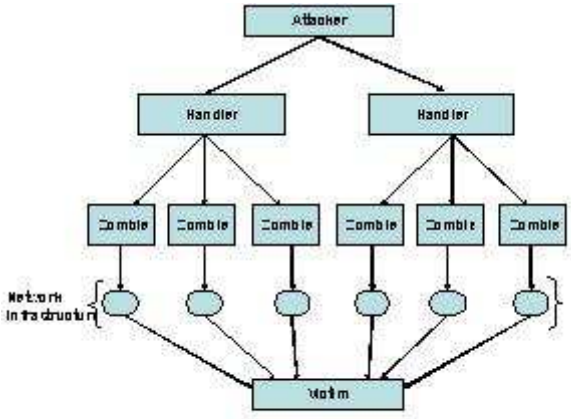


Figure 1. **DDoS Attack.**

The idea presented here is based on the work in [4, 5] that finds the complexity of optimal DDoS attack design for a given graph and distributed application. In this work we describe a two player game played on a physical graph. A computer network is modeled by a directed physical graph structure in which computers are graph nodes and links connecting computers are graph arcs.

The Physical environment (computer hardware infrastructure) is represented by a directed graph structure (*EG*).

$$EG = [EV, EE] \tag{1.1}$$

where *EV* is the set of vertices or nodes (*computers*) with associated computational bandwidth and *EE* a set of directed edges or links (*communications lines*) with associated communications bandwidth. If *EV* consists of *x* computers, it

is expressed as a vector [ev_1, ev_2, …, ev_x] where ev_i is the computational bandwidth available at node *i*. *EE* is expressed as a *n* by *n* connectivity matrix with element ee_{ij} denoting the bandwidth available at the link connecting ev_i and ev_j.

The virtual (Blue) environment is a distributed application consisting of a set of distributed programs. To successfully execute, the distributed programs must be placed on physical nodes from *EG* in a way that lets them communicate with each other at all times. This environment is represented by a "logical" directed graph structure (*BG*).

$$BG = [BV, BE] \quad (1.2)$$

where *BV* is a set of vertices or nodes (distributed programs) and *BE* is a set of edges or links. The capacity of each element of *BV* is the amount of CPU bandwidth it requires. The capacity of each element of *BE* is the amount of network bandwidth required between its associated nodes. If *BV* consists of *y* processes, it is expressed as a vector [bv_1, bv_2, …, bv_y] and *BE* is expressed as an *m* by *m* connectivity matrix with element be_{ij} expressing the volume of communications between bv_i and bv_j.

This work finds connectivity bottlenecks and the set of nodes that are needed for the Blue distributed process to function. We then find the minimum number of zombies and the minimal amount of flow required to attack bottleneck links and essential nodes. Our results show a strong relationship between the connectivity of the graph and the ability of individual network members to resist DDoS attacks. The results obtained can be used to design robust networks, for example by providing a concrete metric for comparing design alternatives.

II. KEY CONCEPTS

Source Node: The node that is the starting point for a data flow is the source.

Sink Node: The node where the flow terminates is the sink.

Max-Flow: In a graph, the max-flow is the maximum volume that can be sent from a given source node to a given sink node [6].

Min-cut: The min-cut is the smallest set of edges that is necessary for the source to send the max-flow to the sink. The removal of these edges from the network graph completely disconnects the source node from a sink node. The removal of any one of these edges reduces the size of the max-flow.

Max-flow min-cut theorem: "The maximal amount of flow is equal to the capacity of a minimal cut [6]."

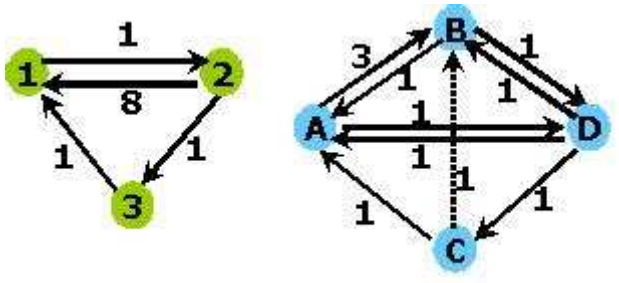
III. PLAYER 1 - BLUE

Player 1 (Blue) is a distributed application on the network which consists of programs executing on physical nodes. They consume CPU resources on the local physical node. Each pair of programs has a known communications bandwidth requirement. These programs must communicate with each other to execute successfully. We first determine the possible computer nodes on *EG* where Blue programs can reside. This gives us a set of "feasible Blue configurations" that is the set of mappings of logical nodes to physical nodes, where the logical graph's CPU and communications needs are satisfied by the physical graph. Blue's goal is to ensure that the Blue - distributed programs remain connected at all times. To find the set of feasible configurations for Blue i.e. the set of mappings of *BV* (distributed program – logical graph) onto *EV* (physical graph) Blue has to satisfy two classes of constraints:

1. Node Capacity Constraints which are not dealt with in this paper. Interested readers are encouraged to refer to [4, 5]. Examples given in this paper ignore the CPU bandwidth.
2. Edge Capacity Constraints which are the subject of this paper.

IV. EDGE CAPACITY CONSTRAINTS

For each element be_{ij} of *BE* connecting elements bv_i and bv_j of *BV* where bv_i (bv_j) is mapped to ev_k (ev_l) the max-flow [6] on *EG* from ev_k to ev_l must be greater than or equal to the bandwidth requirement of be_{ij}. If bv_i and bv_j are on the same node, the value of the max-flow is considered infinite, since no network bandwidth is consumed.



(a) (b)

Figure 1. Connectivity Graphs (a) Physical Connectivity Graph (b) Blue Connectivity Graph

Two blue nodes can be placed on two different physical nodes if the arc/edge capacity (communication requirement) between the two blue nodes is less than or equal to the arc capacity (bandwidth) of the two physical nodes. This check can be expressed as:

Max- Flow between two physical nodes ≥ Arc Capacity of two blue nodes

(1.3)

To determine if the communication requirements between two Blue nodes placed on different physical nodes is satisfied; we need to know the maximum bandwidth available between two physical nodes. To determine this we calculate the max-flow using algorithms from [6].

We use this to construct a set of lists of possible Blue positions on the physical nodes, which we denote as Blue configurations. We need to verify that all outgoing arcs from any given Blue node and all incoming arcs to that same Blue node satisfy all their associated capacity constraints simultaneously without over committing available bandwidth. To verify this, we execute three verifications on the lists of possible Blue configurations:

- *Verification 1*: For every Blue node, check that all outgoing arcs from that Blue node simultaneously satisfy the capacity constraints. This is done by carrying out a row wise connectivity check for all the blue nodes.
- *Verification 2:* Similarly check all incoming arcs to the Blue nodes. This is done by carrying out a column wise connectivity check for all the Blue nodes.
- *Verification 3:* Check if placing combinations of blue nodes on the physical nodes leads to exhausting the arc capacities between the physical nodes.

For these verifications we calculate the min-cut for all the mappings of Blue arcs to physical nodes given by the possible Blue configurations: the min-cut gives the bottleneck edges that constrain a source's ability to communicate to a sink.

V. Feasible Blue Configurations

Final Blue configurations: The positions that satisfy these three verifications give the possible physical locations for Blue nodes. The various combinations of these physical nodes are then checked for consistency with the Blue nodes by verifying if they form a cycle. After forming the Blue configuration, each Blue configuration is cross checked to verify if all the communication capacities available on the physical arcs are sufficient to hold all the Blue nodes at once. If there are *n* such Blue configurations, the set of feasible blue configuration mappings is denoted as:

$BC = \{BC_1, BC_2, \ldots\ldots\ldots, BC_n\}$

(1.4)

For Red to be certain that it can disrupt Blue, it must be able to disrupt all Blue configurations. We now show how Red can find the minimal set of nodes to do so.

VI. Packet Flooding Distributed Denial of Service Attacks

In this attack, packets are sent to flood links and exhaust arc capacities. Table 1 shows Blue configurations and tree diagram we use to illustrate a branch and bound solution as shown in Figure 2. Details on executing branch and bound search can be found in [7,8].

Set of nodes	BC_1	BC_2	BC_3	BC_4
A_1	1-3,2-4	1-3	2-4,1-4	1-4
A_2	2-4	2-4	1-3	
A_3	3-4	3-4,1-4,4-5		

Table 1 Blue configurations and min-cut edges.

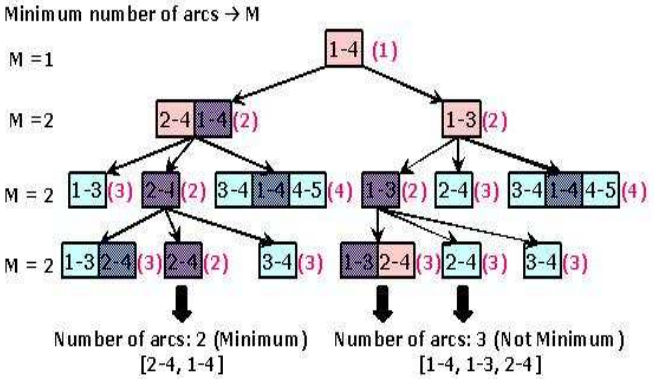


Figure 2. Branch and Bound Search Tree for Arc Attack.

Disabling set A_2 from BC_1, set A_2 from BC_2, set A_1 from BC_3 and set A_1 from BC_4 can disable all the above Blue configurations. Using a Branch and Bound approach, we find the minimal set of edges that we need to disable to be certain that Blue is disrupted.

VII. Zombie Traffic

Now that Red knows the set of edges to disrupt, Red needs to find the amount of flow necessary to disable them. In the Figure 2, arcs (2-4) and (1-4) need to be disabled. Using the following variables:

RT → Red Traffic generated by the zombies

λ packets → Blue (Legitimate) traffic

C → Capacity of the physical arc to be attacked

The total traffic T is:

$$T = \lambda + RT \quad (1.5)$$

Traffic dropped D is:

$$D = (\lambda + RT) - C \quad (1.6)$$

Percentage of Blue (legitimate) traffic in the total traffic P is:

$P = \lambda \div (\lambda + RT)$

(1.7)

Expected rate of Blue (legitimate) traffic loss LTL is therefore:

$LTL = \lambda \div (\lambda + RT)\,[(\lambda + RT) - C]$

(1.8)

The attacker will win i.e. will be successful in flooding the arc when:

$LTL \geq$ *Blue slack Traffic (BS)* (1.9)

Blue Slack traffic (BS) is defined as:

$BS =$ *Capacity – [Blue Flow]* (1.10)

If a Blue arc does not have the available capacity to send the required flow it will try to send this flow on the other arcs in the same min-cut. So we have to check if these other arcs have a slack that can be used to send this flow. Hence, the Blue flow can be given by,

Blue Flow = [Blue Capacity – Slack of other arcs in min-cut]

(1.11)

Therefore:

BS = Capacity – [Blue Capacity – Slack of other arcs in min-cut]

(1.12)

Using equation (1.8) and (1.9) we get,

$BS \geq \lambda \div (\lambda + RT)[(\lambda + RT) - C]$

(1.13)

Blue Slack traffic (*BS*) should be at least equal to LTL. Solving for *RT*,

$$\frac{BS \times (\lambda + RT)}{\lambda} = \left[(\lambda + \mathrm{RT}) - \mathrm{C}\right]$$

$$RT = \frac{C}{\left[1 - (\mathrm{BS} \div \lambda)\right]} - \lambda$$

(1.14)

Equation (1.14) gives us the red traffic an attacker needs to generate in order to disable a Blue arc. Also in the above equation, Blue Slack is assumed to be strictly equal to LTL (1.9) which need not be true. So 'traffic' little more than *RT* i.e. → (*RT* + 1) is needed to disable the arc. Also in the above case *BS* is assumed to be less than λ. If a remainder of zero is obtained i.e. when $BS = \lambda$, then an infinite amount of flow will be needed to disable that arc, making that arc immune to DDoS.

VIII. ZOMBIE PLACEMENT

The vulnerable nodes and arcs and the amount of flow to be routed to the arcs are now known. The final step is to find the source of Red traffic to be generated i.e. Zombie Placement. Steps for finding optimal zombie positions,

1. In every Blue configuration for each physical source and sink we have the max-flow and the min-cut. Calculate *RT* for each min-cut using (1.14). The *RT* is calculated for a single arc and not the entire min-cut. It is very easy to convert the *RT* for a single arc to *RT* for a min-cut. For the capacity *C* use the sum of the physical capacities over all the arcs in the min-cut, λ is Blue traffic over that min-cut. Blue Slack can be a little difficult to calculate. For simplicity, we will ignore Blue interfering with its own traffic. So equation (5.18) becomes,

   BS = Capacity – [Blue Capacity]

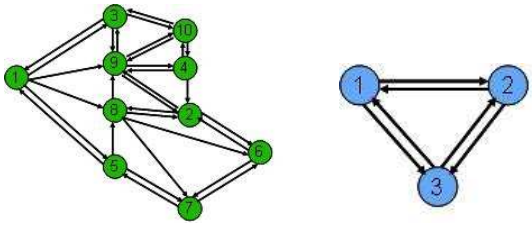
   (1.15)

   where Slack of other blue nodes is assumed to be zero.
2. Consider the physical nodes without the Blue nodes as sources (as we do not want to place the zombies on the same node as the Blue node). Sinks can be all the physical nodes. A good source candidate will be close to the Blue source and a good sink candidate will be close to the Blue sink.
3. Check if the max-flow for any of the sources is greater than RT of any one min-cut in every Blue configuration. For a Blue configuration if any one value of RT > max-flow, select that node.
4. Repeat the above steps for all the Blue configurations.
5. Pick up a common zombie node in all the Blue configurations. If we do not find a single zombie node we have to look for two or maybe more zombie nodes. We use the Branch and Bound search tree [7, 8] to calculate the minimum number of zombie nodes common to all the Blue configurations. The above Branch and Bound search tree has been implemented in C language.

In this way the optimal zombie positions to execute a packet flooding DDoS attack can be determined.

The calculation of Blue configuration and zombie placement is done using MATLAB. SSFNet simulations were used to verify if the Blue configurations were correct and to verify if the zombie places are indeed minimum.

We consider an example of 10 physical nodes, 30 arcs and 3 distributed Blue applications. SSFNet simulations were performed for the example.



(a) (b)

Figure 21. Input Networks (a) Physical Network (b) Blue Network.

The simulation time was set to 1000 seconds. The Blue configuration that was chosen randomly for the simulation was [2, 3, 8]. The minimum number of zombies needed to attack this Blue configuration was placed on physical nodes [1, 9]. The normal traffic starts at 200 seconds. The attack command is given at 250 seconds and the attack traffic starts at 270 seconds and ends at 390 seconds. Both the queue monitor and IP flow monitor were used to collect data statistics. Since we have not incorporated the use of queues for our simulations, the queue length was kept a little low and equal to 50 packets. Figure 3 and 4 are the animation snapshot of the example and the corresponding graphs of statistics collected using the queue monitor. The graphs statistics show the packets sent and the queue length at the attacked router. The requests (attack packets) are sent to the server connected to the router. But the filtering and the attack are evident at the router to which the server is attached. Figure 3 represents the snapshot when the attack has just started on node 2. One can clearly see that three nodes → 2, 3 and 10 are pink in color i.e. there is continuous packet drop on these three nodes. On examining the min-cut sets of arcs, node 3 to node 2 have four arcs in the min-cut set – [3-1], [10-3], [9-4] and [10-4] and then if we look at the packet drops, router 3 and router 10 have packet drops along with router 2 clearly indicating the bottleneck at the arcs connecting these nodes has been exhausted. Figure 4 shows the exceeded queue length of router 2. Similar kind of behavior is seen at router 10 and router 3, though the packet drops at router 3 are much higher. Node 8 to node 2 has one arc in the min-cut set → [3-1] signifying that packet drops should occur at node 3 as node 3-1 will be the bottleneck link. Note that this snapshot is taken at the time where only Node 2 is attacked. So the packet drop on node 3 at this time of the simulation has occurred due to it being on the min-cut arc connecting node 8 to node 2 and not due it being a Blue host.

For verification purposes zombie nodes placed were placed on physical nodes – 5, 6 and 7 that were strongly connected to the Blue sources. These nodes being very close to source do cause flooding and packet drops at the Blue sources they are close to. For example a zombie placed at node 6 causes packet drops at node 8 but fails to affect node 2 and 3. The same scenario is seen when we place a zombie on 7, it only affects node 8. One of the reasons why 1 and 9 are good zombie places is that they are strongly connected to the nodes belonging to the min-cut and the Blue sources.

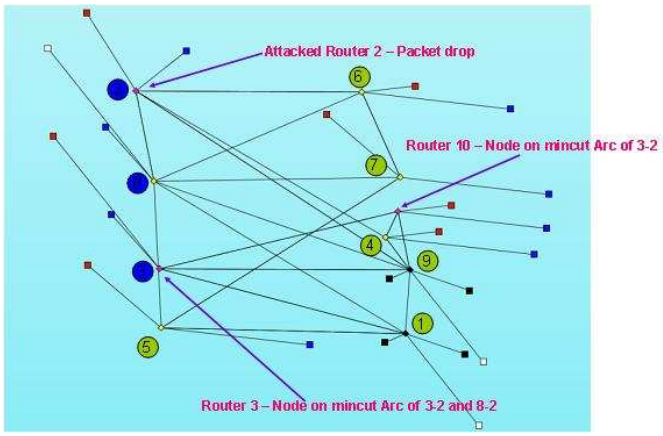


Figure 3: SSFNet simulation – Animation

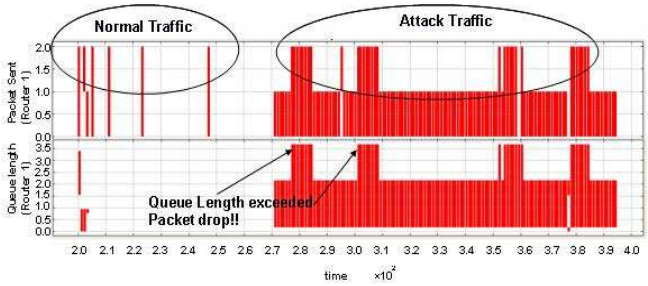


Figure 4: SSFNet simulation - Graphical statistics

IX. CONCLUSION

The DDoS problem is of increasing importance. Many enterprises have been disrupted by this attack. Botnet operators have been known to extort enterprises by threatening DDoS onslaughts. Probably most disturbing is this spring's DDoS attack on NATO ally Estonia, presumably from Russia.

Distributed Denial of Service (DDoS) attacks are particularly difficult to counteract, since the vulnerability they exploit is the open nature of the Internet Protocol. This paper presents an initial step towards serious DDoS prevention, in that it shows how to develop optimal DDoS attacks. We suggest using the minimal number of zombies required to disable a system as a metric for DDoS vulnerability. Distributed system implementations that require more zombies to disable are naturally more resilient to disturbance by DDoS.

Much work remains to be done. A step in the right direction is the approach in [5] which attempts to develop methods for strategically reconfiguring distributed systems in response to DDoS attacks.

REFERENCES

[1] CERT Coordination Center, Denial of Service Attacks [Online]. Available: http://www.cert.org/tech_tips/denial_of_service.html.

[2] S. Dietrich, N. Long, and D. Dittrich, "Analyzing Distributed Denial of Service Attack Tools: The Shaft Case", in *Proceedings of the LISA 2000 System Administration Conference*, December 2000, New Orleans, LA.

[3] L. Garber, "Denial of Service Attacks Rip the Internet," *Computer*, vol. 33, no.4, pp. 12-17, April 2000 [Online]. Available: (http://www.fas.org/irp/congress/1992_cr/s920624-spy.htm) Last visited 12/12/2006.

[4] R. R. Brooks, *Disruptive Security Technologies with Mobile Code and Peer-to Peer Networks*, Boca Raton, FL: CRC Press, 2005.

[5] Chinar Dingankar, *Enterprise Security Analysis Including Denial of Service Countermeasures*, M.S. Thesis, Holcombe Department of Electrical and Computer Engineering, Clemson University, August, 2007.

[6] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, *Network Flows*, Prentice Hall, Upper Saddle River, NJ, 1993.

[7] J. Clausen, "Branch and Bound Algorithms - Principles and Examples," Department of Computer Science, University Copenhagen, [Online]. http://www.imm.dtu.dk/~jha/

[8] E. L. Lawler and D. E. Wood, "Branch and bound methods: A survey," *Operations Research*, vol. 14, pp. 699–719, 1966.

Program Analysis For Sensing Mutant Cyberweapons
(Extended Abstract)

Andrew Walenstein, Arun Lakhotia, Anshuman Singh
Center for Advanced Computer Studies
University of Louisiana at Lafayette

Guna Seetharaman
Air Force Institute of Technology
Wright Patterson Air Force Base, OH

Abstract

To sense the presence of a cyberweapon directly it is generally necessary to either observe its behavior or examine the weapon itself in the form of a program. The latter requires some type of content or program analysis. Automated methods for generating program mutants promises to make those types of analyses extremely challenging. For the past five years we have researched program analysis methods that can be used to improve the effectiveness of cyberweapon sensors. One of the aims common to much of our research is to discover ways of countering the problems introduced by automated program mutation techniques. Four threads from this research are reviewed. The review is used to support an argument that program analysis will be an important part of cyberweapon sensor construction and, further, that problems faced in program analysis relate to issues of sensor placement and data collection.

1 Introduction

The history of program analysis research in computing science is long, and many elegant techniques have been created for parsing, modeling, and transforming programs, and for proving various properties. As a research area, program analysis has much to say about examining code to see what it is, what it does, and how it works. Therefore one might expect to see program analysis research appear frequently in cyberweapon sensing research. Including, say, work related to sensor placement and information fusion.

Yet in searching the literature it does not appear that program analysis and reverse engineering work appears to relate strongly to the typical concerns of information fusion and sensor management. Is there really not much for the the fields to say to each other? Or is the situation temporary and these these links should become stronger in the future? In this position paper, we try to present a case for the latter. Specifically, we argue that: (a) program analysis research is relevant to cyberweapon sensor construction, and (b) that research in sensor placement and information fusion may have important things to say about program analysis in that context.

The first part of the argument draws upon our experiences in research aiming to improve malicious program detectors, which we treat as a type of sensor. Focus is placed on the problems introduced by the automated mutation techniques an attacker may employ. Automated mutation systems can create variants of a cyberweapon. Four of our research efforts for countering such attacks are summarized. We argue that such mutation techniques pose important challenges to attack sensing and, as a results, one can expect program analysis to be an important component in future sensor construction.

For the second part we present a case that fundamental research on sensor placement and information fusion could become important contributors to future work in program analysis in cyberweapon sensor systems. In the past, we have argued that classic program analysis techniques for analyzing benign code are not well-suited for analyzing code created by adversaries [11]. We argued that, in the future, classic program analysis will change in order to better respond to the demands of "adversarial software analysis." This paper adds to this the argument that cyberweapon analysis will be enhanced by integrating dynamic and static analysis. Critically, we argue that the integration will require distributing cyberweapon sensing in time and place, and that such a shift will require careful understanding of issues of sensor placement and then fusing information from distributed sources to make a sensing decision.

The case for the importance of program analysis in

cyberweapon sensor construction is made in Section 2. Section 3 presents an argument for why the issues of sensor placement and information fusion may be critical components of building future cyberweapon sensors.

2 Program analysis for sensors

Certain classes of cyber attack involve the execution of the attackers code on remote computers. In order to stop the execution one needs to sense that an attack is in progress. One of the methods attackers can use to make it more difficult to sense the attack is to generate modified versions of the cyberweapon. For example, they could use some type of code obfuscation transformation [2]. In addition, they could some type of system that automatically generates weapons.

This section presents an argument for why program mutation techniques are likely to mean that program analysis will be an important part of sensor construction in the future. Four threads of research on countering the effects of program mutation are then reviewed with an aim to anticipate future relationships between program analysis and sensor management and information fusion.

2.1 Research on program variation

For over five years, the Software Research Laboratory at the University of Louisiana at Lafayette has been investigating the problem of analyzing malicious programs, such as cyberweapons. Below is a distilled overview of four of the research threads that focus on providing methods for dealing with program variations. That is, these are methods that may be used to sense mutant cyberweapons. The review will avoid dwelling on details of the actual program analyses—the reader is referred to the papers themselves—and will instead aim to highlight the significance of the analyses for mutant cyberweapon sensing, and will draw out unsolved problems that each approach raises.

2.1.1 Semantics-based behavior matching

A given program function may be written in many different ways. This makes it impossible to completely anticipate how a given attack may be implemented. One of our early efforts in system security aimed to provide a way of defining and searching for high-level behavior patterns of malicious activity. The hope was to be able to use these could form the basis of building sensors that can detect specific forms of malice however they are implemented. We made use of the sophisticated static analysis techniques of *model checking* [8]. Model checking permits one to define possible execution properties of programs and, then to search for a possible execution sequence that violate those properties—no matter how they are actually implemented.

The advantage of such program analyses is that the behavior matching is performed at a *semantic* level, meaning that variations in program form are made irrelevant. There are, however, several problems faced with implementing such approaches. For one, the analyses can be expensive. Moreover, the high-level analyses require precise and high-level program models. For instance our approach needed an accurate control flow graph (including system calls), which in turn requires accurate disassembly. Defeat either disassembly or control flow graph extraction and the whole sensing system is defeated [7, 11].

2.1.2 Program normalization

Normalization is the process of removing unwanted variation in input to make processing easier. If programs could be normalized, then the complicating changes introduced by mutations can be removed. In our early work in this area we used standard program transformations common to optimizing compilers [6]. This approach removes variation but cannot guarantee a single normal form. In later work we used the theory and mechanisms of a program analysis paradigm called "term rewriting" [12, 13]. Here the relevant theory could precisely define classes of mutation engines that could be "neutralized."

The advantage of such analyses is that they can fairly efficiently neutralize the effects of certain mutation engines. Some of the problems include the necessity of modeling the attacker's transformation system. The performance, while probably orders of magnitude faster than model checking, may also be too slow for many applications, and the technique is vulnerable to the same sorts of attacks on the disassembly phase of the analysis.

2.1.3 Approximated matching

When automated mutation appeared in viruses, one of the first responses of the anti-virus industry was to increase the power of their matching technology by enabling them to define more abstract—yet precise—patterns [9]. We sought to introduce approximated techniques that can anticipate certain classes of mutation in the match. To this end, we adapted information retrieval techniques typically used for matching queries to documents [4]. Specifically, we introduced a new type of match term we called "n-perms"; these enabled programs to be matched even if the ordering of their operations were modified. Sensors could then be built on top of the approximate matching capability such

that mutated versions of known weapons can be recognized [14].

The advantage of such analyses is that, unlike generic character-based approaches such as n-grams over binaries, they attempt to take into account (a) the structure and content of programs in defining the comparison features, and (b) the types of program mutations that may underlie the differences. The approach has shown to be fairly scalable even without typical optimization schemes [14], but it still is sensitive to attacks on the disassembly phase of the analysis (though the approximated nature of the match means it is less susceptible than more precise approaches).

2.1.4 Statistical signatures

Defense against powerful mutation techniques often requires powerful program analysis techniques. Many of the analysis approaches proposed in the research community—including our own—can be expensive to run, making them poorly suited in uses where suspect programs are scanned before being used. We sought a fast approach for filtering out programs that are unlikely to be mutants generated automatically. We proposed an approach based on statistical inference about the probability of a program being output from a given mutation system [1]. The essential idea is borrowed from linguistic studies of authorship analysis: mutation engines normally have idiosyncrasies in the sort of output they generate, and these idiosyncrasies can be detected by examining the occurrence of various program constructs.

Such program analyses are useful in that they can rapidly steer sensors away from useless computation. However the approach gains much of its speed at the expense of precision in that it may fail to provide much decision-making help in non-obvious cases. In addition, the problem of correct disassembly is also faced.

3 Distributed cyberweapon sensing

We have argued that program analysis techniques can offer many possibilities for cyberweapon sensing. In reviewing the problems of applying them, however, certain problems are still being faced. These include: cost problems (computation space or time), and the brittleness of algorithms in the face of various programming attacks. In a previous position paper [11] we presented an argument that the latter problem can be expected because classic program analysis is poorly matched to the challenges of analyzing code created by an adversary. We argued that in the future analysis methods may be able to become more *robust* by shifting to processing methods similar to those found in, say, speech processing, which are forced to to deal with with noisy or inconsistent data, and imprecise results.

One of the best examples of a shift in this direction comes from the paper by Kruegel *et. al* [5]. It presents a technique for disassembly that is more robust in the presence of obfuscations—which is a problem we noted was being faced by many sensing approaches based on program analysis. Classic program analysis is strictly staged, with disassembly coming before control flow extraction. It also requires precise and correct results from each stage: correct and complete disassembly is required by the control flow modeling algorithms. Kruegel *et. al* persuasively argue that obfuscation makes this approach infeasible, and describe a disassembler built with different processing principles. Their disassembler does not produce precise results to begin with, as it starts with an initial approximation (guess) and that approximation is refined as more knowledge is acquired. Moreover, the processing is not strictly staged; rather, an initial disassembly is created, an initial control flow graph is created, and this graph is used to refine the disassembly. The process incrementally works towards the correct solution. This effort thus matches our proposal that program analysis methods are likely to become more opportunistic and incremental in their processing, and employ more approximate or soft representations.

Our previous position did not consider *dynamic* program analyses. Traditionally, dynamic analyses are integrated with static by letting one type improve the approximation created by the other type. For example Udupa *et. al* [10] use procedure call tracing (dynamic technique) to improve the accuracy of statically constructed call graphs. We argue here that in applying program analysis to cyberweapon construction, dynamic and static analyses may need to be integrated, and that the integration will entail solving issues of sensor placement and information fusion.

In this context, a sensor may consist of a patched system call that tracks the calling context and updates an approximated call graph. Another type of sensor might be a programmed hard drive controller that senses changes to protected system files. Other sensors could be typical traffic sensors at the level of a firewall in the node's network interface or on a router or network appliance. The challenge is to ensure that the sensor information can be utilized to improve the models generated, in part, through static analysis.

References

[1] M. R. Chouchane, A. Walenstein, and A. Lakhotia. Statistical signatures for fast filtering of instruction-substituting metamorphic malware. In *WORM '07: Pro-*

ceedings of the 2007 ACM workshop on Recurring malcode, pages 31–37, New York, NY, USA, 2007. ACM.

[2] C. Collberg, C. Thomborson, and D. Low. A taxonomy of obfuscating transformations. Technical Report 148, Department of Computer Science, The University of Auckland, July 1997.

[3] W. G. Griswold and D. Notkin. Automated assistance for program restructuring. *ACM Transactions on Software Engineering and Methodology*, 2(3):228–269, 1993.

[4] M. E. Karim, A. Walenstein, A. Lakhotia, and L. Parida. Malware Phylogeny Generation using Permutations of Code. *Journal in Computer Virology*, 1(1-2):13–23, November 2005.

[5] C. Kruegel, W. Robertson, F. Valeur, and G. Vigna. Static Disassembly of Obfuscated Binaries. In *Proceedings of the 13th USENIX Security Symposium*, pages 255–270. USENIX, 2004.

[6] A. Lakhotia and M. Mohammed. Imposing Order on Program Statements and its implication to AV Scanner. In *Proceedings of 11th IEEE Working Conference on Reverse Engineering*, WCRE, pages 161–171, Los Alamitos, CA, 2004. IEEE Computer Society Press.

[7] A. Lakhotia and P. K. Singh. Challenges in getting 'formal' with viruses. *Virus Bulletin*, pages 15–19, September 2003 2003.

[8] P. K. Singh and A. Lakhotia. Static verification of worm and virus behavior in binary executables using model checking. In *Proc. of the 2003 IEEE Workshop on Information Assurance*, pages 298–300, 2003.

[9] P. Szor. *The Art of Computer Virus Research and Defense*. Symantec Press. Addison Wesley Professional, 1st edition, 2005.

[10] S. K. Udupa, S. K. Debray, and M. Madou. Deobfuscation: Reverse engineering obfuscated code. In *12th. IEEE Working Conference on Reverse Engineering*, 2005.

[11] A. Walenstein and A. Lakhotia. Adversarial software analysis: Challenges and research. In *Proceedings of the Workshop on Code Based Software Security Assessments (CoBaSSA)*, Nov 2005.

[12] A. Walenstein, R. Mathur, M. R. Chouchane, and A. Lakhotia. Normalizing metamorphic malware using term-rewriting. In *Proceedings of the Sixth IEEE International Workshop on Source Code Analysis and Manipulation (SCAM 2006)*, pages 75–84, Philadelphia, PA, Sept. 2006. .

[13] A. Walenstein, R. Mathur, M. R. Chouchane, and A. Lakhotia. Normalizing metamorphic malware using term-rewriting. *Journal in Computer Virology*, 2007. (to appear).

[14] A. Walenstein, M. Venable, M. Hayes, C. Thompson, and A. Lakhotia. Exploiting similarity between variants to defeat malware. In *Proceedings of BlackHat Briefings DC 2007*, Mar. 2007.

A Robust Clustering Algorithm for Target Tracking in Wireless Acoustic Sensor Networks

Raghu Kisore Neelisetti*, Alvin Lim*, Pratima Agrawal† and Qing Yang*
*Department of Computer Science and Software Engineering
Auburn University, Auburn, Alabama 36849
Email: [neelira,lim,yanqin]@auburn.edu
†Department of Electrical and Computer Engineering
Auburn University
Auburn, Alabama 36849
Email: agrawpr@auburn.edu

***Abstract*—The advancement of MEMS technologies has made it possible to produce tiny wireless sensor devices. These tiny sensors hold the promise of revolutionizing sensing in a wide range of application domains because of their flexibility and low cost. One such application is target localization and tracking using acoustic signal of the target. The capabilities of these tiny devices are limited by their battery power, storage capacity, computational power and communication bandwidth. These limited capabilities make the decisions made by each sensor error prone. Hence most target detection and tracking algorithms require the sensors to work in groups in order to improve the reliability of target tracking algorithms. This makes it necessary for deployed sensors to discover and group together so that their coverage can be maximized. In addition, with the advent of video sensor networks it has become possible to record a video of the target once it is detected and later be relayed to an external agent. In this paper, we propose a clustering algorithm that tries to produce the maximum number of possible clusters given a certain type of deployment. The proposed clustering algorithm is distributed in nature and has the ability to reconfigure in the event of node failure. The algorithm is highly localized and hence does not need flooding across the entire network. Since the algorithm allows for more clusters to track the same region the system reliability is greatly improved.**

I. Introduction

The increasing capabilities and declining cost of computing and communication devices, has led to an increase in the number of applications of wireless sensor networks. One such application is battlefield surveillance. Surveillance involves both detection and tracking of intruders. Tracking based on the strength of acoustic signal received by a set of sensors is a common technique. This requires deployed sensors to work in groups. Though each sensor is capable of detecting the presence of a target, its results are error prone and could result in false alarms. Hence to increase the accuracy of the detection algorithm it becomes necessary to fuse the measurements of a group of sensors. This makes it necessary for the deployed sensors to work in small clusters so that the overall reliability of the surveillance system can be improved.

Wireless sensors can be used to detect various features such as thermal signatures, ferro-magnetic content or acoustic signal. The absence or presence of a target phenomenon can be inferred by aggregating the measured values from a small group of sensors deployed. In this paper we assume that each sensor is equipped with a microphone and hence can record the acoustic signal.

Detection requires that the system discriminate between a target's absence and presence. Successful detection requires a node to correctly estimate a target's presence while avoiding false detections in which no target is present. This can be done by using triangulation based on acoustic measurements made by at least three sensors. On the other hand target tracking is more complicated since it involves maintaining the target's position as it moves over time in a region covered by the sensor network's field of view. Therefore the tracking algorithm should be able to identify the orientation of the target's path and its velocity in addition to location of the target's position. One such algorithm to track the target is the CPA (Closest Point of Approach) algorithm [1]. In this algorithm a group of four sensors make CPA measurements, and then, based on the measurements, the trajectory of the target can deduced with reasonable accuracy. The capability of the application can be further enhanced if we assume that each sensor is equipped with a camera. This is possible with the advent of low cost cameras that are capable of providing resolution in the order of mega pixel. The CPA measurements made by each of the four sensors will be reported to a node (actually one of the four sensors) where the target's trajectory can be computed. Once the parameters of the target's trajectory are computed, the camera associated with the sensor can be programmed to pan in the target's direction. However, this requires real time data from each of the four individual sensors, i.e. the CPA measurements generated by the sensors must be delivered to a node in a timely fashion. Out-dated reports are of little use. These timing constraints call for a clustering algorithm.

The overall system architecture consists of two self-contained components: the acoustic target tracking subsystem which deals with the detection and processing of acoustic signals and the communication subsystem which is responsible for exchanging sensor data and high quality tracking results. One way to address the limited computational and battery power of wireless sensor devices is to organize the sensors

into clusters. Sensors in each cluster coordinate in sensing and communication to perform the sensing task. To deal with the inaccuracy in measurement and unreliability typical of low-end devices in remote or hostile environments, we suggest a clustering algorithm that organizes the sensors into redundant clusters so as to obtain more robust results.

In general, target classification and tracking algorithms rely on information provided by a cluster of sensors. In case of target classification each sensor is equipped with different modalities, such as magnetic, radar, thermal, acoustic, chemical, electric, seismic and optical. Hence the target classification draws its results from observations made by a cluster of modalities. This emphasizes the need for a clustering algorithm that can exploit the redundancy in the sensor deployment and reduce the latency in the exchange of raw data and the amount of raw data that needs to be exchanged.

The proposed clustering algorithm is distributed in nature and the number of clusters to be formed can be easily controlled. Further, since the cluster head chooses its member nodes from its one hop neighbors, the raw data has to travel only one hop. Finally, the target tracking results of each cluster head can be progressively fused with those of its neighboring clusters.

The remainder of the paper is outlined as follows. Section II reviews related work; Section III defines the problem, while Section IV provides details of the proposed clustering algorithm, followed by performance evaluation in Section V. Finally, Section VI offers some concluding remarks.

II. Related Work

The system that we are presently referring to is an intrusion detection system which is essentially a surveillance situation of practical importance and is well-suited to wireless sensor networks. The intrusion detection system is designed as a dense, distributed, wireless network of multi-modal, resource-poor sensors combined into loosely coherent sensors that perform in situ detection and estimation. There are several issues of interest in designing such distributed intrusion detection systems. The first and foremost is the sensor deployment algorithms. These algorithms aim at maximizing the field of coverage of a given set of sensors. One metric to identify the effectiveness of a deployment strategy is by measuring the worst and best case coverage paths. In [2] the authors optimize deployment of heterogeneous sensors through Linear Programming. In [3] the authors propose three approximation algorithms for a variation of the SET K-COVER problem, where the objective is to partition the sensors into covers such that the number of covers that include an area, summed over all areas, is maximized. In [4] the authors analyze the minimum number of nodes needed for random deployment so as to meet a desired value for least path of exposure metric. They assume Gaussian distribution for the random deployment strategy. In [5] the authors propose algorithms to provide k-coverage in a mostly sleeping network. The aim of the algorithm is to save energy and at the same time provide certain desired degree of coverage of the protected region at all times. However, all these algorithms analyze the degree of coverage from the perspective of target detection but not target tracking.

One method of judging the effectiveness of a particular sensor deployment algorithm is by measuring the worst and best case coverage. [6], [7] provide algorithms to measure the worst and best case coverage based on Voronoi diagrams. In [8] the authors analyze worst case coverage (also called the breach path) in case of directional field-of-view sensor networks.

Line in the sand [9] system is a prototype model that can detect and classify up to three different target types. In [9], the authors discuss various issues in developing such systems, largely emphasizing data fusion algorithms. VigilNet [10] is a real time large-scale sensor network system that can track, detect and classify the targets in a timely and energy-efficient manner. In [10], the authors perform mathematical analysis of various delays and accuracy of the system. Both the above systems rely on mutual co-operation of group of clusters. They both assume the availability of a clustering algorithm. In [11], a target detection algorithm localizes a sound source using triangulation based on the acoustic measurements made by a group of three sensors. Once again the existence of clustering algorithm is assumed.

[12] evaluates three different architectures for fusing data collected by the sensors. The three schemes analyzed are a centralized scheme, a progressive scheme and a distributed scheme. A centralized source number estimation scheme is a processing structure in which all sensors send their raw data to a central processing unit where source number estimation is performed. A progressive source number estimation scheme is a processing structure that a group of sensors update the source number estimation result sequentially based on each sensor's local observation and the partial estimation result from its previous sensors in the sequence. So, the information transmitted through the network is the estimation result or partial decision. Finally, a distributed or cluster based source number estimation scheme is a structure including two levels of processing: source number estimation within each cluster and decision fusion between different clusters. The authors conclude that the cluster-based distributed approach using the progressive intra-cluster estimation has the best performance in the sense that it can provide much higher detection probability than the centralized approaches, while at the same time occupying the least amount of network bandwidth and consuming the least amount of energy. [13] Introduces Markov chain Monte Carlo data association algorithm to track an unknown number of targets. The algorithm once again relies on the existence of a clustering algorithm.

The clustering algorithm presented in this paper has the ideal features pointed out in [12]. The algorithm is distributed in nature and allows for intra cluster data aggregation. The intra cluster data aggregation is made possible by the overlapping nature of the clusters. This also leads to redundancy and increases the success rate of target detection.

III. Problem Definition

Data fusion algorithms rely on clustering algorithms so that the raw data collected by individual sensors can be combined in an efficient way. One such data fusion algorithm useful for target tracking application is based on closest point of approach or CPA algorithm. The CPA algorithm [1] estimates the target motion parameters based on CPA measurements made by at least four sensors. The target motion parameters being speed, direction, bearing of the target's path and the precise location of the target at a certain instance of time.

The purpose of clustering algorithm is to fuse the CPA measurements made by individual sensors at minimum cost (in terms of energy) and at the same time provide best coverage possible for a given deployment. One way to achieve data fusion is to deliver the CPA measurements to a centralized location. However transmission of raw data to the centralized location would mean transmitting 4 raw data packets instead of one fused value. This is not economical in terms of energy. Instead, clustering allows for local data fusion. Since each clusterhead is only one hop away from its member nodes less energy is consumed in transmission of raw data and also the probability of loosing raw data is reduced.

A cluster once formed can track a target only with certain accuracy. The primary sources of error are false alarms at each sensor, loss of raw data and sensor failures. Hence it is necesary to provide redundancy. Redundancy can be achieved by allowing more than one cluster to track the same region. In the proposed algorithm we allow for redundancy by allowing two clusters to share a predetermined number of nodes. By increasing the number of shared nodes more clusters are formed in the same region and hence increses the system reliability. However Figure 5 shows that failure rate can be considerably reduced by increasing the number of clusters tracking a region from 1 to 3. However this increased reliability comes at an extra cost in terms of energy. In the proposed algorithm reliability can be controlled by adjusting the number of nodes two clusters can share. This also leads to a series of overlapping clusters and allows for data fusion between inter clusters.

A. Closest point of approach

Each sensor monitors the acoustic signal from the target with the help of a microphone, i.e. it monitors the signal energy for a given time window. The sensor confirms the presence of a target (called event) once the signal strength exceeds a certain threshold. The threshold is dynamically updated based on background noise statistics to reduce false alarm rate. Once a node detects an event (i.e. the presence of a moving vehicle), it stores a time series segment corresponding to the event. Figure 1 shows the time series segment corresponding to the interval in which the energy first exceeds the threshold (start of event) and eventually drops below the threshold (end of event) after reaching a peak value. The time at which the acoustic signal peaks is called the closest point of approach (CPA).

The CPA measurements made by each of the four sensor are reported to a centralized location (which is just any of the four sensors) where the individual measurements are fused together by the CPA algorithm to determine the target motion parameters, such as precise location at a certain instant of time, velocity and orientation. Once these parameters have been calculated, the camera attached to the sensor can be programmed to record a video of the target and the recorded video can be sent to an external actor by using a data centric routing protocol, such as Directed Diffusion. However, the algorithm has pitfalls. There are certain configurations in which the algorithm fails to estimate the motion parameters. Figure 2 shows all the possible target trajectories with respect to the way sensors are deployed.

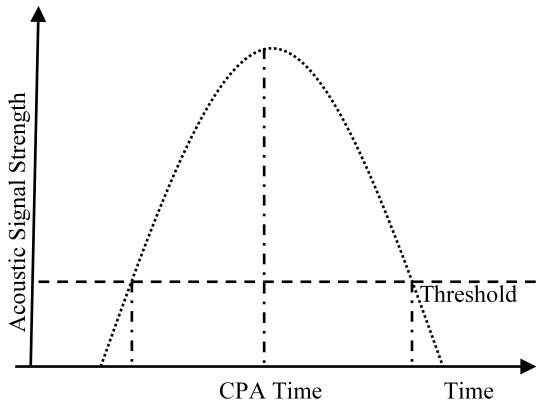


Fig. 1. Event detection by thresholding the energy of the acoustic signal detected by the microphone. The horizontal line represents the threshold. The maximum reading corresponds to CPA time.

The CPA algorithm can estimate the target parameters only when there is uneven number of sensors deployed on either side of the target's trajectory. At the minimum, to detect the target motion parameters we need CPA measurements from four sensors with 3 on one side and one on the other side of the target trajectory. Also the algorithm requires the three sensors that are on one side of the trajectory to be non co-linear. In all other cases the solution is ambiguous [1].

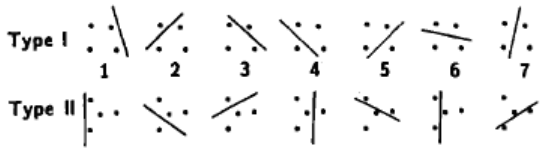


Fig. 2. Classification of target trajectories according to the way of sensor field decomposition.

The clustering algorithm described in this paper avoids the issue of even deployment of sensors by grouping five sensors together into one cluster. One of the five sensors is chosen as the cluster head. All the member nodes send their raw data to the cluster head, which then makes use of the CPA algorithm to estimate the target motion parameters. Figure 3 shows a cluster formed between five nodes and the dotted lines

represent the possible paths a target can take. Each cluster will have a cluster head to which all the other sensors can send in their results. The cluster head then runs the CPA algorithm to identify the trajectory's parameters. The cluster head needs CPA measurements from only 4 sensors, but does not know which 4 measurements will lead to a solution and hence tries out combinations. It then ignores the invalid combinations and averages the valid solutions.

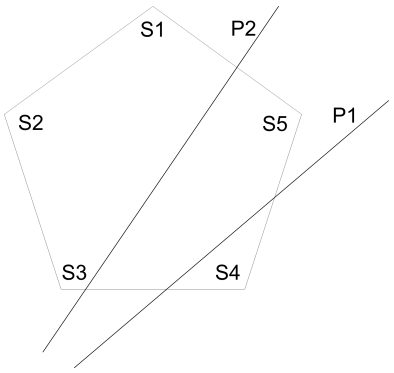


Fig. 3. A cluster formed with five sensors.

Each cluster has a certain failure rate in detecting the target. The failures arise because of the inefficiency of the individual sensors in detecting CPA time. The errors could be because of the ambient noise or because of the failure to identify the right threshold. Further, in some cases the raw data packets sent by some of the sensors to the cluster head might be lost. Hence it becomes necessary to have redundant clusters so that the tracking efficiency can be improved.

Assuming that each sensor fails to detect the event with a probability p, then the probability of failure for a cluster to track an event can be obtained by evaluating the following two cases.

Case 1: Assume that the target takes the path P1 i.e. 1 sensor on one side of the target trajectory and all the other sensors on the other side. Let P1 represent the probability of failure.

$$P = 1 - probability\ of\ success = 1 - ({}_4C_3 * (1-p)^4 * p + (1-p)^5) \quad (1)$$

Case 2: Assume that the target takes the path P2 i.e. 2 sensors on one side of the trajectory and the rest on the other side. Let P2 represent the probability of failure.

$$P = 1 - probability\ of\ success = 1 - ({}_2C_1 * (1-p)^4 * p + (1-p)^5) \quad (2)$$

Since there are only 5 possible scenarios in which Case 1 can happen and 10 possible scenarios in Case 2 can happen, the probability with which a cluster can fail to detect an event is 0.33*(1) + 0.67*(2).

Figure 4 represents the individual failure rates of each case and the overall failure rate with changing failure rates of individual sensors. A sensor fails to detect because of various factors, such as ambient noise and issues associated with the fine tuning of thresholds for the acoustic detector. The total

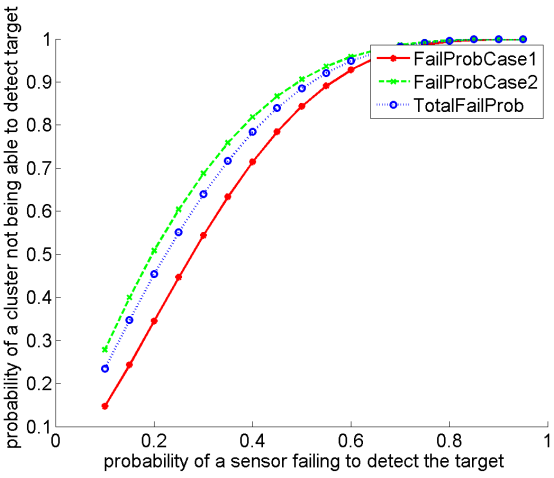


Fig. 4. Failure probability of a cluster in detecting the target given individual sensor failure probability.

success rate can be improved by having more than one cluster monitor a certain region. This makes it necessary to have overlapping clusters. If we have six sensors, then we can have six clusters such that any two clusters differ by at least one sensor. Incorporating new nodes and forming new clusters with different set of sensors can give a different perspective to evaluate the tracking parameters. Figure 5 shows how the total failure to track a target decreases as more clusters track the target. It can be seen that the failure probability can be reduced greatly by increasing the number of clusters tracking a target from 1 to 3. Any further increase in the number of clusters does lead to a significant improvement.

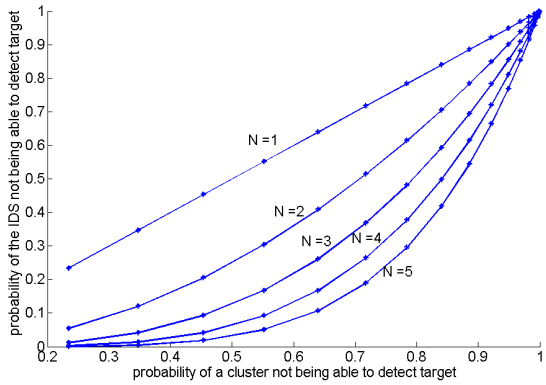


Fig. 5. Improvement in the target detection as more number of clusters monitors a region.

In the algorithm presented in the next section, the amount of redundancy can be controlled by controlling the number of sensors two clusters can have in common.

B. Protocol Overview

The clustering algorithm forms as many polygons as possible and prevents polygons with exactly the same set of sensors

from forming. Ties are broken by giving preference to the sensor with lower ID. Once formed, a cluster head remains in cluster head state until it detects that one of its member nodes is not responding. The cluster head node and the member nodes poll one other with HELLO messages to detect node failure. In the event of node failure, the cluster head disbands the cluster and moves to the initial state and starts all over again. Each sensor (cluster head) manages only one cluster at a time, but can be a member node to any number of clusters. The overlapping nature of the clusters improves the reliability of the system as the same portion of the field is monitored by more than one cluster. An overview of the protocol is shown in Figure 6.

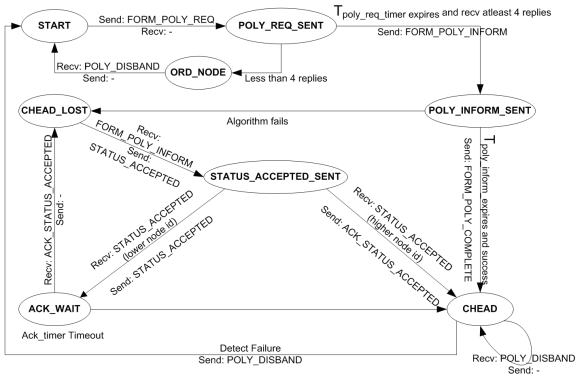


Fig. 6. State transition diagram of the clustering algorithm.

The protocol has eight states and makes use of seven messages. The states are START, CHEAD, HELLO, POLY_REQ_SENT, POLY_INFORM_SENT, and CHEAD_LOST. A brief description of each state is given below.

- START: This is the initial state of the node soon after it has been deployed out on the field.
- CHEAD: A node moves to this state on successfully forming a polygon. It receives CPA information from the member nodes and runs the CPA algorithm to determine the target parameters. It periodically sends out HELLO messages to verify that its member nodes are alive. A node in CHEAD state remains so until one of its member nodes dies.
- POLY_REQ_SENT: A node interested in forming a cluster broadcasts a FORM_POLY_REQ message and move to this state while it waits for replies from nodes that interested in forming a cluster.
- POLY_INFORM_SENT: Once a node decides on its member nodes, it broadcasts a FORM_POLY_INFORM message. This message carries the ID of all the member nodes.
- CHEAD_LOST: when a node that intends to form a polygon realizes that a similar polygon is being formed by another node (with a smaller ID), it gives up its attempt to form a polygon and moves to CHEAD_LOST state.

The seven messages are FORM_POLY_REQ, FORM_POLY_REPLY, FORM_POLY_INFORM, FORM_POLY_COMPLETE, STATUS_ACCEPTED, ACK_STATUS_ACCEPTED, HELLO, and POLY_DISBAND.

- FORM_POLY_REQ: this is a request message broadcast by a node that intends to form a cluster soliciting replies from nodes that are interested in joining a cluster.
- FORM_POLY_REPLY: Upon receiving a FORM_POLY_REQ message, a node interested in joining a cluster replies with a FORM_POLY_REPLY message.
- FORM_POLY_INFORM: A node that has decided on the member nodes of its cluster broadcasts a FORM_POLY_INFORM. This message contains the IDs of the member nodes. The purpose of this message is to avoid duplicate clusters. Duplicate clusters are those clusters that have identical nodes.
- FORM_POLY_COMPLETE: This is a broadcast message sent by a node to inform its member nodes about the successful formation of the cluster. The sender of the message becomes the cluster head. The member nodes send raw data about any target they detect to the cluster head.
- HELLO: The cluster head and the member nodes make sure that the cluster is intact by periodically exchanging HELLO messages between them.
- POLY_DISBAND: Once a cluster head concludes that one of its member nodes is not responding, it disbands its cluster by broadcasting a POLY_DISBAND message. It then moves to START state and starts all over again.

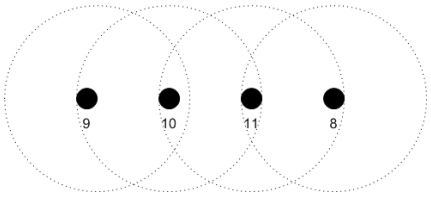


Fig. 7. A special case of the clustering algorithm.

In the rest of the section, we explain the purpose of STATUS_ACCEPTED and ACK_STATUS_ACCEPTED message. The main purpose of the message is to reconfigure the clusters in the event of node failures and at the same time ensure that neither too many redundant clusters are formed nor too few clusters are formed. Too few clusters might lead to void region. Void region is the region that is not monitored by any cluster.

- STATUS_ACCEPTED: In Figure 7, assume that nodes 9, 10 and 11 all try to form clusters at the same time. Also assume that the set of member nodes of 9 and 10 differ by a single node and the set of members of 10 and 11 also differ by a single node. However, assume that the set of member nodes of clusters being formed by nodes 9 and 11 differ by 2 nodes. By virtue of lower ID, node 9 gets to form a cluster while 10 moves to CHEAD_LOST

state. Assume that each of the nodes have exchanged the FORM_POLY_INFORM messages. By the nature of the node positions, node 9 is aware of the cluster being formed by 10; 10 is aware of the clusters being formed by both 10 and 11; and 11 is aware of the cluster being formed by 10. Now there is a tie between nodes 9, 10 and 11. Since the algorithm allows for nodes with lower ID to form clusters, only node 9 would be able to form a polygon. This because 10 would loose the race to 9 and 11 would loose the race to 10. We could better monitor the field if 9 and 11 can form clusters as their respective set of member nodes differ by 2 nodes. Hence, in such situations, 10 allows 11 to form a cluster by sending a STATUS_ACCEPTED message.

Before the STATUS_ACCEPTED message reaches node 11, assume that node 11 looses to node 8 (since 8 has lower ID, it has precedence over 11 in case of, tie.). In this case, we would end up with just two clusters formed by nodes 9 and 8. This might lead to a region between nodes 9 and 8 not being monitored. In order to avoid such situations, node 11 sends out a STATUS_ACCEPTED message to node 10, and node 10 goes ahead to form a cluster. Node 11 will then move to STATUS_ACCEPTED_SENT state.

- ACK_STATUS_ACCEPTED: While in STATUS_ACCEPTED_SENT state, if a node receives a STATUS_ACCEPTED message from a node with a lower ID, it replies with a STATUS_ACCEPTED message and waits for (ack_timer period) the lower ID to acknowledge the message with ACK_STATUS_ACCEPTED message. On the other hand, if it receives an STATUS_ACCEPTED message from a higher node ID, it moves to CHEAD state and the lower ID acknowledges with a ACK_STATUS_ACCEPTED message. This final step of the algorithm assures that nodes with a lower ID gets to form a polygon even in the presence of wireless losses, and, at the same time, assures that no void regions (regions that are not monitored by any cluster) are formed.

IV. Performance Evaluation

The clustering algorithm was implemented in ns-2.27 [14]. The total number of protocol packets generated in order to maintain the clusters for a period of 100 seconds was measured. Four different deployment topologies were evaluated. The first was grid deployment in which 100 nodes were laid in 10 * 10 matrix and separated by 40m. The second topology was a random deployment consisting of 150 sensors uniformly deployed over a 670*670 m field. The next two sensor deployments were based on pentagonal tiling.

Pentagonal tilings were analyzed since it is possible to have non-overlapping clusters of five sensors and hence assure that each part of the field can be monitored by at least one cluster. In an ideal case where each cluster can monitor the region that it encloses with a probability of 1. By having one of the five nodes of each pentagon to be a cluster head and the other

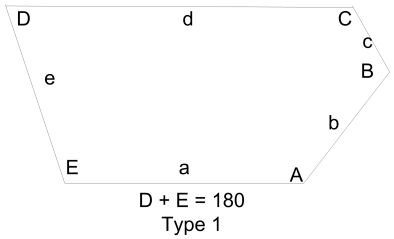


Fig. 8. Pentagonal tessellations.

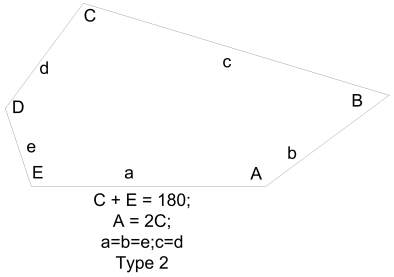


Fig. 9. Pentagonal tessellations.

nodes as its member nodes, the entire field can be monitored with a minimum number of clusters. In all, 14 different types of convex pentagons [15] can tile a plane but we evaluated only two of them. Figure 8 and 9 shows the geometrical properties of the two.

In tessellation of Type I, 280 sensors were deployed in a rectangular field of 500 * 1000m. In case of Type II, a total of 300 nodes were deployed in a rectangular field of 500*1000m.

A node that intends to form a cluster can adopt different strategies in identifying its member nodes. Two different strategies were evaluated. In the first strategy the node intending to form a cluster chooses the nodes that are closest to it, while in the second version the algorithm chose the farthest nodes to form a cluster. By choosing the farthest nodes the region tracked by each cluster is maximized, but increases the protocol overhead to maintain them as they tend to be less stable. On the other hand by choosing closest nodes as member nodes leads to more stable clusters but reduces the area monitored by each cluster. In each scenario the algorithm was analyzed by allowing the cluster heads to form clusters with varying number of identical nodes (the same nodes appear in two different clusters). The number of identical nodes was chosen to be 2, 3 and 4.

Choosing the farthest nodes increases the area tracked by a single cluster but also increases the protocol overhead when compared to the minimum coverage approach. This is because the member nodes of a cluster and the cluster head use HELLO messages (for the simulations the hello timer was set to 4 sec) to verify that all the members of the cluster are alive. A node that receives a HELLO message replies with a HELLO_REPLY message, both of which are broadcast messages. All the member nodes that receive the broadcast

message update their hello timer for the node that just sent the HELLO_REPLY. However, when the cluster head chooses the farthest nodes as its member nodes there is a higher probability that not all the member nodes are within one another's communication range and this leads to increased hello messages.

The policy of allowing clusters to have more and more shared nodes leads to an increase in the number of clusters and increase in redundancy. As the clusters are allowed to have more shared nodes, it can be seen that the protocol overhead is reduced. This is because when N=2 (N being the number of identical nodes), two clusters that are in close vicinity have a greater chance of having more than two nodes in shared (i.e. identical). Hence the sensors spend more time in resolving ties and this leads to increased protocol overhead. For the same reason the clusters face lesser conflicts with neighbours when N=3 and N=4. In case of the maximum coverage approach, the overhead is more or less uniform for N=2, 3, 4 as each cluster is formed with farthest nodes there is a lesser chance of conflict. This leads to an increase in the number of clusters formed. Also, more clusters would mean more HELLO packets to make sure that the member nodes are alive. Furthermore, in the maximum coverage approach since the member nodes of a cluster are geographically more distant from each other; the ability of a cluster to track a target is reduced.

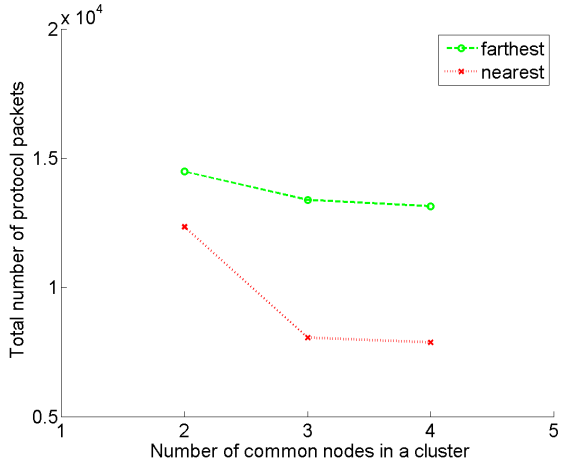


Fig. 10. Protocol overhead in Grid deployment.

From Figures 10, 11, and 12 it can be seen that both the number of identical nodes two different clusters is allowed to have and the approach taken to choose the member nodes while forming a new cluster affect the protocol overhead.

The probability of conflicts between clusters is also dependant on the deployment strategy. In random and tessellation based deployment strategy it can be seen that for the case N=2 the minimum coverage approach spends even more time in resolving conflicts whereas the protocol overhead for the maximum coverage approach is constant for all the deployment patterns. From Table I, it can be seen that more clusters

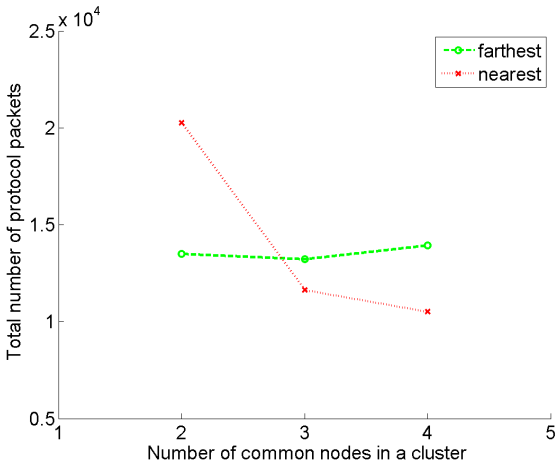


Fig. 11. Protocol overhead in Random deployment.

TABLE I
COMPARES THE TOTAL NUMBER OF CLUSTERS FORMED IN CLOSEST APPROACH AND THE FARTHEST APPROACH. READ AS (MINIMUM COVERAGE VS MAXIMUM COVERAGE)

	N = 2	N = 3	N = 4
Grid	50 Vs 83	72 Vs 93	92 Vs 99
Random	48 Vs 98	77 Vs 135	112 Vs 139
Type I Tessellation	90 Vs 202	192 Vs 272	260 Vs 280
Type II Tessellation	74 Vs 238	135 Vs 286	155 Vs 288

can be formed by choosing the maximum coverage approach. However this leads to increase in number of cluster heads and hence increase in the number HELLO message exchanged to make sure that the cluster is active.

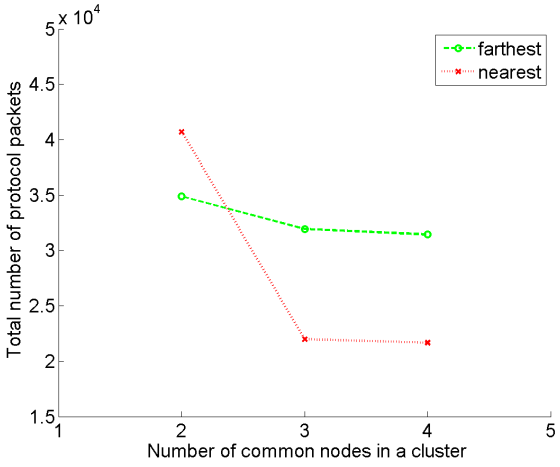


Fig. 12. Protocol overhead in case of sensor deployment using Type 1 pentagonal tessellation.

Whenever a node dies the cluster heads of all the clusters that the dead node belongs to are dissolved and a race for

forming new clusters begins. Also since a node, can be a cluster head and at the same time be a member node of another cluster, the dissolution of one cluster could lead to a ripple effect. A larger ripple means a larger number of clusters have to be reorganized and hence more protocol overhead. The extent of the ripple is dependent on the geographical location of the node and its relationship to other nodes.

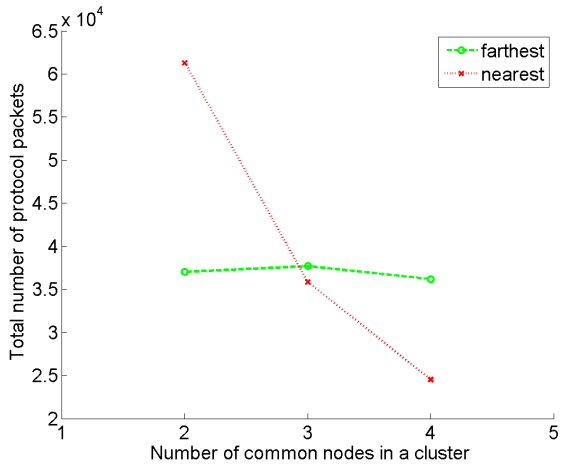


Fig. 13. Protocol overhead in case of sensor deployment using Type 2 pentagonal tessellation.

So in each of the deployment scenarios we kill a node that is in the center of the field. In Figures 14-17 it can be seen that the ripple is contained to a smaller region, and hence, the increase in protocol overhead is small. In the proposed clustering algorithm, a cluster head once in CHEAD state continues to remain so until one of its member node dies. This limits the spread of ripple once a node dies. We analyze the protocol overhead when the clusters are formed using minimum coverage approach. The additional protocol packets generated to re-organize the clusters in the event of node failure is maximum for grid deployment and insignificant in all other deployment strategies.

Table 1 shows that by adopting the maximum coverage approach (each cluster head forms a cluster by choosing geographically farthest nodes from its one hop neighbors) in the clustering algorithm we can form as many as twice the number of clusters with closest approach (each cluster head forms a cluster by choosing geographically closest nodes from its one hop neighbors) even when N=2. This justifies the large protocol overhead observed in case of maximum coverage approach.

V. Conclusion

In this paper we propose a distributed clustering algorithm to track intruders. The localized nature of the algorithm ensures that reconfiguration does not significantly increase the protocol overhead. The algorithm builds a series of overlapping clusters which allow for more than one cluster to track a region. This redundancy improves the overall system

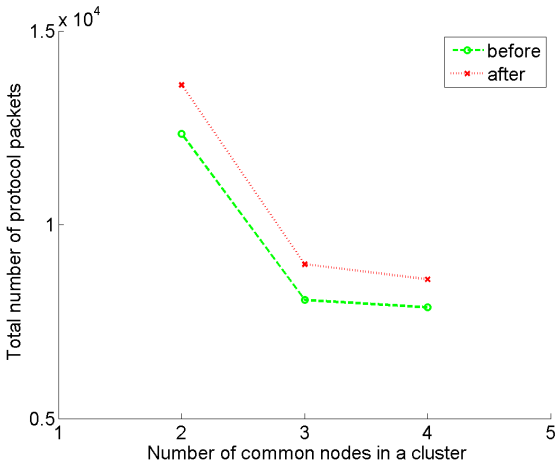


Fig. 14. Protocol overhead in case of Node failure in grid deployment (minimum coverage approach).

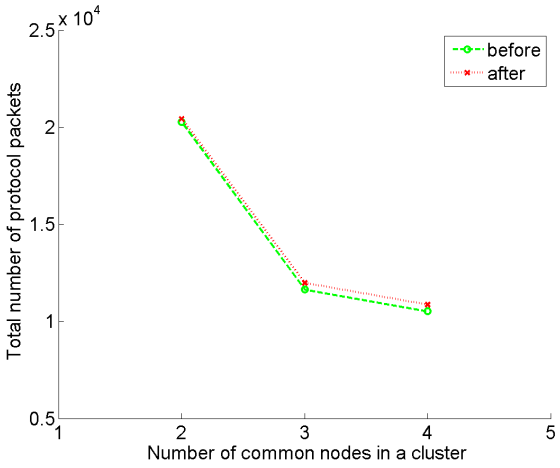


Fig. 15. Protocol overhead in case of Node failure in Random deployment (minimum coverage approach).

reliability. The overlapping clusters also allow for tracking of curvilinear targets. We analyzed the factors that affect the protocol overhead. The protocol overhead depends on the sensor deployment strategy, number of clusters and the approach adopted by the cluster head in choosing its member nodes. Maximum coverage approach forms larger numbers of clusters than the minimum coverage approach and also leads to proportionate increase in the protocol overhead. Node failures lead to re-organization of clusters and hence increase protocol overhead. In the worst case there is a 11% increase in protocol overhead to reconfigure the clusters. This happens when the sensors are deployed in a grid. In the future we would like to analyze the efficiency of clustering algorithms by measuring the average probability of tracking.

References

[1] F. Dommermuth, "The estimation of target motion parameters from cpa time measurements in a field of acoustic sensors." in *J. Acoust. Soc.*

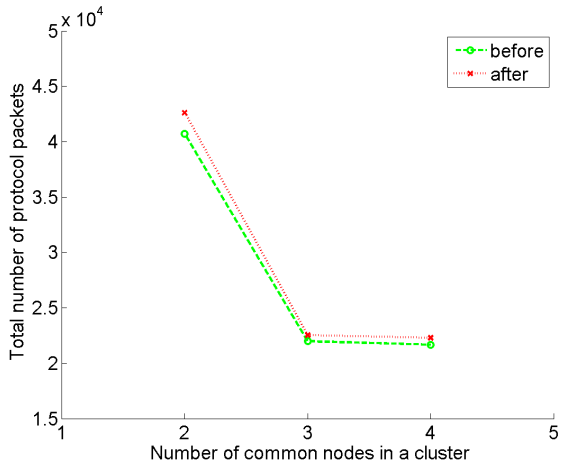


Fig. 16. Protocol overhead in case of Node failure when sensors are deployed using Type 1 pentagonal tessellation (minimum coverage approach).

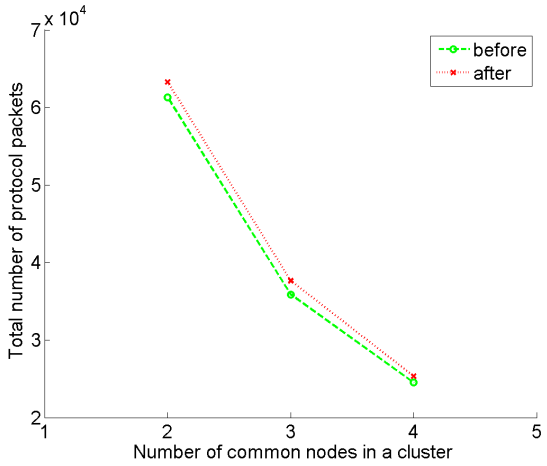


Fig. 17. Protocol overhead in case of Node failure when sensors are deployed using Type 2 pentagonal tessellation (minimum coverage approach).

Am., vol. 83, no. 5, 1988, pp. 1476–1480.

[2] X. Xu and S. Sahni, "Approximation algorithms for sensor deployment," in *Innovations and Real-Time Applications of Distributed Sensor Network, 2006. Proceedings. 2nd International Symposium on*, 2006.

[3] Z. Abrams, A. Goel, and S. Plotkin, "Set k-cover algorithms for energy efficient monitoring in wireless sensor networks," in *IEEE IPSN*, 2004.

[4] T. Clouqueur, V. Phipatanasuphorn, S. K., and P. Ramanathan, "Sensor deployment strategy for detection of targets traversing a region," in *Mobile Networks and Applications*, vol. 28, no. 8, 2003, pp. 453–461.

[5] S. Kumar, T. Lai, and B. J., "On k-coverage in a mostly sleeping sensor network," in *Mobile Computing and Networking, 2004. Proceedings. Tenth Annual International Conference on. ACM*, 2004.

[6] G. Veltri, Q. Huang, G. Qu, and M. Potkonjak, "Minimal and maximal exposure path algorithms for wireless embedded sensor networks," in *SenSys*, 2003.

[7] S. Megerian, F. Koushanfar, M. Potkonjak, and M. Srivastava, "Worst- and best-case coverage in sensor networks." in *Mobile Computing, 2004. IEEE Transactions on*, vol. 3, no. 4. IEEE, 2004.

[8] J. Adriaens, S. Megerian, and M. Potkonjak, "Optimal worst-case coverage of directional field-of-view sensor networks." in *Sensor, Mesh and Ad Hoc Communications and Networks, 2006. The third annual IEEE Communications Society Conference on*. IEEE, 2006, pp. 336–345.

[9] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita, "A line in the sand: A wireless sensor network for target detection, classification, and tracking." vol. 46, no. 5. Computer Networks, 2004, pp. 605–634.

[10] H. Tian, A. Pascal, Y. Ting, L. Liqian, Z. Gang, S. Radu, C. Qing, A. Stankovic, and T. Abdelzaher, "Achieving real-time target tracking using wireless sensor networks." in *Embedded Computing System, 2007. ACM Transactions on*, 2007.

[11] W. Qixin, C. Wei-Peng, Z. Rong, L. Kihwal, and S. Lui, "Acoustic target tracking using tiny wireless sensor devices." in *Information Processing in Sensor Networks, 2003. Proceedings. 2nd International Workshop on*, 2003.

[12] W. Xiaoling and Q. Hairong, "Mobile agent based progressive multiple target detection in sensor networks." in *Acoustics, Speech, and Signal Processing, 2004. Proceedings. IEEE International Conference on*, 2004.

[13] S. Oh, P. Chen, M. Manzo, and S. Sastry, "Instrumenting wireless sensor networks for real-time surveillance." in *Robotics and Automation, 2006. Proceedings. the International Conference on*, 2006.

[14] "Network simulator 2 (ns2)," University of California at Berkeley, 1997. [Online]. Available: http://www.isi.edu/nsnam/ns/

[15] B. Grunbaum and G. Sheperd, *Tilings and Patterns*. New York, U.S.A: W.H Freeman And company, 1986.

On Energy-Efficient Priority-Based Routing in Heterogeneous Sensor Networks

Min Meng, Yunyue Lin, Yi Gu, Qishi Wu, Nageswara S.V. Rao

Abstract— **Sensor networks are becoming increasingly pervasive in various applications ranging from mission-critical military operations to environmental surveillance to industrial process monitoring. In single-sink homogeneous sensor networks, traditional routing algorithms are mainly focused on selecting a path with a minimum number of hops or a minimum amount of transmission energy cost. One fundamental problem associated with these routing algorithms is that the energy of the sensors near the sink or on the critical paths is depleted much faster than that of other nodes, resulting in unbalanced energy consumption. In the worst case, the failure of one or more critical nodes may break down the entire sensor network. We present the system architecture of heterogeneous sensor networks that consist of a small number of high-end sensors serving as sink nodes and a large number of low-end sensors responsible for environmental sensing and data collection. Within the framework of heterogeneous sensor networks, we propose a path bottleneck-oriented and energy cost-based routing algorithm that compares multiple paths to multiple sinks in path selection. A multi-objective function considering both path bottleneck and energy cost is constructed to optimize the balance of network-wide energy consumption. Extensive simulation results show that the proposed routing algorithm outperforms traditional routing algorithms in prolonging the lifetime of sensor networks.**

Keywords: Path bottleneck, energy-efficient routing, sensor network lifetime, heterogeneous sensor network

I. Introduction

Wireless sensor networks are becoming increasingly pervasive in many military, civil, agricultural, and industrial applications ranging from mission-critical homeland security defense, battlefield assessment, health care improvement, environment surveillance, climate research, disaster recovery, ecological forestry and wildlife monitoring, to manufacturing process control. In most sensor network applications, especially those deployed in large-scale harsh environments, sensor nodes are powered by irreplaceable batteries with limited energy supply to provide basic sensing, communication, and computing functions. Optimizing the balance of network-wide energy consumption to prolong the lifetime of sensor networks has been a challenging but vital task in sensor network implementation.

Previous research efforts on sensor networks are mainly focused on homogeneous sensor networks, where all sensor nodes have the same capabilities in terms of sensing, communication, processing, and energy supply. However, extensive studies have shown that there exists a fundamental limit on application performance and system scalability in homogeneous sensor networks. In particular, recent research has demonstrated the performance bottleneck of homogeneous sensor networks both theoretically and through simulation experiments and testbed measurements [16], [10]. In homogeneous sensor networks with a single sink, various routes used by sensor nodes for data transmission converge to the same destination, and therefore the sensor nodes near the sink or on the critical paths consume more energy and deplete faster than other nodes. The traditional routing algorithms in single-sink sensor networks usually target minimizing the number of hops or the total energy consumption incurred on the path from a source node to the sink. In such routing schemes, a selected path that consists of a minimum number of hops or consumes a minimum amount of energy may very likely include nodes with relatively low residual energy, which could be rapidly exhausted by more subsequent

Meng, Lin, Gu, and Wu are with the Department of Computer Science, University of Memphis, Memphis, TN 38152, Email: {mmeng, ylin1, yigu, qishiwu}@memphis.edu. Rao is with the Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, Email: raons@ornl.gov. Please send all correspondence to Wu.

intensive data transmissions. In the worst case, the failure of one or more critical nodes may destroy the usability of the entire sensor network.

Assuming that the sink nodes have sustainable energy supply, balancing the energy consumption among the sensor nodes is critical to prolong the lifetime of the network. Toward this end, we present the system architecture of heterogeneous sensor networks (HSNs) that consist of a small number of powerful high-end sensors (H-sensors) serving as sink nodes and a large number of low-end sensors (L-sensors) responsible for environmental sensing and data collection. HSNs can significantly improve sensor network performance and also facilitate network configuration and management. Within the framework of HSNs, we propose an energy-efficient path bottleneck-oriented (PB) and energy cost-based (EC) routing algorithm (PBEC) that compares multiple paths from a source node to multiple sinks in path selection. Here, the path bottleneck is defined as the minimum amount of data transmission that can be supported by the residual energy of any node on the path. Based on PB and EC routing algorithms, we construct a multi-objective function taking into account both path bottleneck and transmission energy cost to optimize the balance of network-wide energy consumption. We implement the proposed PBEC algorithm and present performance comparison with other two routing methods. Extensive simulations illustrate that the proposed PBEC algorithm achieves superior performance in prolonging the sensor network lifetime over traditional routing algorithms based on energy cost.

The rest of the paper is organized as follows. In Section II, we conduct a broad survey on related work on sensor network architectures and routing algorithms. In Section III, we present the system architecture of HSNs. The PBEC algorithm is proposed and discussed in Section IV. In Section V, we implement the PBEC algorithm and compare its performance with that of the existing methods. We conclude our work and discuss future research directions in Section VI.

II. Related Work

Sensor networks formed by a large number of densely deployed sensor nodes have been used in a wide spectrum of fields [3] for monitoring habitats [25], pollutants [34], [11], and environments [35], [21]. Due to the well recognized shortcomings of homogenous sensor networks, recent research efforts have been focused on a new sensor network architecture, namely Heterogeneous Sensor Networks (HSNs), and a great deal of research is being carried out on energy-efficient routing [24], [23], sensor node deployment [26], and various network lifetime issues [40]. The network topology and data collection [8] are mainly determined by the system architecture of sensor networks, which has been studied in a limited scope for multi-sink sensor networks.

In the design of sensor networks, the search for an efficient and fault-tolerant architecture is a very important task because the sensor network performance is critically dependent on its interconnection topology. Two types of traditional network architectures, namely the committee organization and the hierarchical organization, are discussed by Wesson *et al.* in [38]. The JIK network proposed in [20], [31] has a mixed structure such that the leaf nodes in the network are organized as many complete binary trees while the roots of which are completely connected. Since the integration errors of the lower nodes in the JIK network accumulate as the information goes up the hierarchy, it is difficult to identify the faulty component of the network. This problem was solved by interconnecting the nodes at every level of the JIK network as a de Bruijn network [19], resulting in a new versatile architecture referred to as the binary multi-level de Bruijn network (BMD) [19]. The BMD has shown several fault tolerant properties so that using it as a basis in the network design makes the network tolerant to node or link failures.

Several recently deployed sensor network systems including those deployed in Great Duck Island, Maine [36] and James Reserve, California [1] utilized different types of sensor nodes and followed an HSN model. The studies on various aspects of HSNs also appeared in a number of literatures. In [14], Girod *et al.* consider an HSN consisting of small MICA2 sensor nodes as well as more powerful Personal Digital Assistants (PDAs). The Center for Embedded Networked Sensing at UCLA [12] developed EmStar – a software environ-

ment for HSN research [13]. The Embedded Networks Lab at USC [4] developed Tutornet – a tiered sensor network testbed with real sensor nodes [15]. In [32], Rhee *et al.* developed Millennial Net, which provides hardware and software architecture with an assumption of the presence of heterogeneous node energy and communication capabilities. Two real sensor networks are presented in [7] and [37] that utilize heterogeneous nodes for processing and transport tasks. Particularly, Cristescu *et al.* demonstrated that the MICA sensor, which uses very little power but performs complex calculations slowly, is best suited to sensing, while the PDA, which consumes significantly more power but performs computations relatively quickly, is best suited for data fusion [7]. In [41], Yarvis *et al.* studied some design issues in a special type of HSN, where some of the sensor nodes are line powered and have unlimited energy supply, and all other nodes are just one-hop away from the line powered nodes. Line powered sensors may be possible in industry plants, but may not be realistic in many other sensor network applications such as battlefield surveillance and wild animal tracking.

Current sensor data routing protocols fall into two main categories, namely table-driven (proactive) and source initiated on-demand (reactive) routing [33]. Proactive sensor data routing maintains up-to-date routing tables that provide a path from each node to every other node in the network. Destination-Sequenced Distance-Vector Routing (DSDV) is such a routing scheme based on the Bellman-Ford algorithm [29]. Babel uses a feasibility condition that guarantees the absence of loops and attempts at making DSDV more robust and more efficient [2]. Reactive routing runs on the demand of a source node to a destination node, thus the overhead for control packet transmission is tremendously reduced compared with the former. Ad hoc On-Demand Distance Vector (AODV) is a reactive protocol using sequenced distance vector [28], where the source node broadcasts a connection request and other nodes forward this message and record its sender. Routes are backwards confirmed hop-by-hop once the request message reaches a node that already has a route to the destination (or the destination itself), and the route with the least number of hops is selected for data transmission.

Data-centric routing protocols include flooding, gossiping, SPIN [30], directed diffusion [18] and rumor routing [5]. In flooding, a sensor node broadcasts packets to all its neighbors until the destination is reached, while in gossiping, a sensor node sends packets to a randomly selected neighbor. SPIN is a protocol with three-way handshake: ADV, REQ, and DATA. In directed diffusion, each node performs aggregation and caching to achieve good energy efficiency and low delay. Rumor routing is a trade-off between query and event flooding. Heinzelman *et al.* proposed Low Energy Adaptive Clustering Hierarchy (LEACH), an energy-efficient hierarchical routing protocol [17].

All of these routing protocols have their own functionality advantages and performance limitations. The proposed PBEC routing protocol considers multiple paths to multiple sinks and explores a novel concept of path bottleneck in optimizing the balance of energy consumption to achieve a longer network lifetime.

III. System Architecture of Heterogeneous Sensor Networks

In this section, we present a Heterogeneous Sensor Network architecture that can be applied to many practical sensor network applications. In this HSN model, both H-sensors or sink nodes and L-sensors or leaf nodes are powered by batteries and have limited energy and communication capability, but H-sensors have more computing power and higher energy capacity than L-sensors. L-sensors use multi-hop communications to reach H-sensors and H-sensors use multi-hop communications to reach the base station or gateway.

Different from single-sink sensor networks, in HSNs, multiple H-sensors are deployed as sink nodes at strategically selected locations. A large number of L-sensors are deployed in the region of interest to collect environmental measurements. L-sensors connect with each other (neighbor nodes within the radio range) through wireless communications to form an ad-hoc network. The main components of HSNs include task manager, gateway, H-sensors and L-sensors, which are illustrated in Fig. 1.

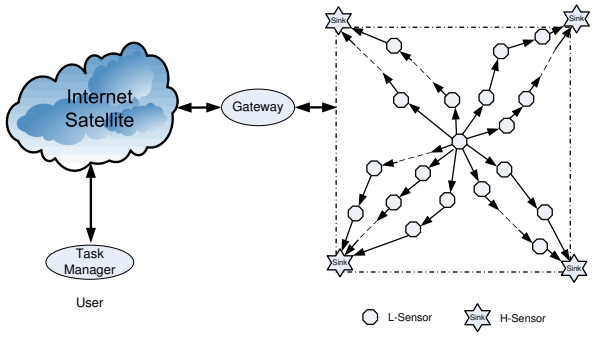


Fig. 1. Architecture and components of heterogeneous sensor networks.

In a typical sensor network application, the user creates queries through a task manager, which are transmitted to a gateway over Internet connections or satellite links. The gateway is responsible for dispatching and gathering data by broadcasting or peer-to-peer communication. An L-sensor selects one of the sink nodes and sends measurement data to it via a multi-hop path. Multiple paths from a source node to a specific H-sensor can be explored by certain route discovery mechanisms similar to the forward broadcast and backward confirmation process in AODV. The general routing objective is to select the best path for data transmission that optimizes the balance of energy consumption in the network. However, the more paths we consider in path selection, the more computation overhead it will incur.

The advantages of HSNs compared with single-sink sensor networks exist in the fact that multiple H-sensors (i) avoid the breakdown of the entire sensor network due to a single sink node failure and (ii) relieve the unbalanced energy consumption among sensor nodes. In HSNs, when an H-sensor fails, the data can be transmitted through other H-sensors. In single-sink sensor networks, all the measurement data has to travel through the sensor nodes near the sink, which, hence deplete energy much faster than other nodes. The data transmission burden in HSNs is shared among all the H-sensors so as to balance the energy consumption and prolong the lifetime of the network.

Typically, the sensor nodes are widely dispersed in the entire area of interest and the H-sensors are deployed in the margins of the area. Multiple paths can be found from each sensor node to every H-sensor. Depending on the locations of sensor nodes and event occurrences, sensor nodes collect measurement data of different sizes and transmit them over different distances to different sinks, which may result in unbalanced energy consumption after running the sensor network for a period of time. In Section IV, we discuss in detail the routing algorithms that target balancing the energy consumption among sensor nodes to prolong the network life time.

IV. Path Bottleneck-Oriented and Energy Cost-Based Routing in HSN

We construct the mathematical models for sensor network, routing path, link communication cost, path energy cost, node energy level, and path bottleneck to facilitate the description of the proposed routing algorithm.

A. Mathematical models

- Sensor network
  A sensor network is depicted as an undirected graph $G(V,E)$, where V denotes the set of all network nodes including a small subset of sinks (H-sensors) V_H and a large subset of L-sensors V_L, and E denotes the set of all direct communication links between any two L-sensors or between an L-sensor and an H-sensor:
  $$V = V_L \cup V_H,$$
  $$E = \{(u,v)\,|u,v \in V_L\} \cup \{(u,v)\,|u \in V_L, v \in V_H\}. \quad (1)$$
- Routing path
  A sensor data routing path is defined as an ordered set of nodes starting from the source sensor node v_{L_1} through one hop or multiple hops to one of the sink nodes v_{H_i}:
  $$P_{(v_{L_1},v_{H_i})} = (v_{L_1}, v_{L_2}, ..., v_{H_i}). \quad (2)$$
  Note that path P can be also considered as a set of component links.
- Link communication cost
  Link communication cost is defined as the energy cost of a direct communication over a

wireless link $L_{(i,j)}$ between two sensor nodes:

$$LC(L_{(i,j)}) = k \cdot d_{(i,j)}^{\alpha} + \tau, \quad (3)$$

where $d_{(i,j)}$ is the physical distance of a direct link $L_{(i,j)}$ between two sensor nodes v_i and v_j, k is the power dissipation parameter of transmission circuit, α is a power dissipation exponent, whose value varies according to the environment, and τ is the total energy cost incurred for sampling, computation, and receiving of sensing data.

- Path energy cost
  The energy cost of a path $P_{(v_L,v_H)}$ is defined as the total energy cost of a data transmission incurred over all component links of path P from a sensor node v_L to a sink node v_H:

$$EC(P_{(v_L,v_H)}) = \sum_{all\ L_{(i,j)}\ of\ P} LC(L_{(i,j)}), \quad (4)$$

- Node energy level
  The network lifetime does not only depend on the amount of energy consumption for data transmission but also the network-wide balance of energy consumption. To account for the latter, we introduce a new concept, *energy level*, which represents the communication capacity of a sensor node v_i on a path $P_{(v_L,v_H)}$ and is defined as the amount of data transmission over a link $L_{(i,j)}$ from the sensor node v_i to its immediate neighbor node v_j along the path P that can be supported by the current residual energy $E_{res}(v_i)$:

$$EL(v_i | L_{i,j} \in P_{(v_L,v_H)}) = \left\lfloor E_{res}(v_i)/LC(L_{(i,j)}) \right\rfloor. \quad (5)$$

- Path bottleneck
  The bottleneck of a path P is defined as the minimum node energy level on the path:

$$PB(P) = \min_{v_i \in P}(EL(v_i)). \quad (6)$$

  Note that the path is broken if its bottleneck decrements to zero.

We now present the routing algorithms based on the path energy cost, path bottleneck, and both.

B. EC: energy cost-based routing algorithm

Suppose that the sensor node v_s sends data to one of n sink nodes in V_H deployed in the network. The total energy cost of a data transmission over a path $P_{(v_s,v_k)}$ from the source node v_s to the sink node v_k is $EC(P_{(v_s,v_k)})$. The EC routing algorithm is described as:

$$\min_{v_k \in V_H}(EC(P_{v_s,v_k})). \quad (7)$$

The pseudo-code for the energy cost-based routing algorithm is shown in Algorithm 1.

Algorithm 1 EC: energy cost-based routing
Input: source node v_s and its routing table (RT)
Output: the label of the selected H-Sensor

```
Select a sink node v_{H_1} ∈ V_H;
MinCost = EC(P_{(v_s,v_{H_1})});
HSensorID = v_{H_1};
for all v_k ∈ V_H, v_k ≠ v_{H_1} do
   Find a path P_{(v_s,v_k)} in RT;
   if EC(P_{(v_s,v_k)}) < MinCost then
      MinCost = EC(P_{(v_s,v_k)});
      HSensorID = v_k;
   end if
end for
return HSensorID.
```

C. PB: path bottleneck-oriented routing algorithm

PB is a routing algorithm based on the bottleneck of a path. We first calculate the path bottleneck from the source node v_s to each H-sensor v_k in the sink set V_H, and then choose the path with the maximum path bottleneck for data transmission:

$$\max_{v_k \in V_H}(PB(P_{(v_s,v_k)})) = \max_{v_k \in V_H}(\min_{v_i \in P_{(v_s,v_k)}}(EL(v_i))). \quad (8)$$

The pseudo-code for the path bottleneck-oriented routing algorithm is shown in Algorithm 2.

D. PBEC: path bottleneck-oriented and energy cost-based routing algorithm

PBEC is a hybrid routing algorithm that considers multiple paths from the source node to a sink node and takes both the energy cost and path bottleneck into account to achieve more balanced energy consumption. The energy cost of data transmission over a path $P_{(v_s,v_k)}$ from source node v_s to sink node v_k is $EC(P_{(v_s,v_k)})$ and the bottleneck of the path

Algorithm 2 PB: path bottleneck-oriented routing
Input: source node v_s and its routing table (RT)
Output: the label of the selected H-Sensor

PBL = path bottleneck list;
MNEL = minimum node energy level;
for all paths $P_{(v_s,v_k)}$ from v_s to $v_k \in V_H$ in RT **do**
  MNEL = $EL(v_s|v_s \in P_{(v_s,v_k)})$;
  for all nodes $v_i \in P_{(v_s,v_k)}$, $v_i \neq v_s$ **do**
    if $EL(v_i|v_i \in P_{(v_s,v_k)}) <$ MNEL **then**
      MNEL = $EL(v_i|v_i \in P_{(v_s,v_k)})$;
    end if
  end for
  PBL[k] = MNEL;
end for
HSensorID = v_{H_1};
MPB = maximum path bottleneck;
MPB = PBL[1];
for all elements in PBL with index i, $i \neq 1$ **do**
  if PBL[i] $>$ MPB **then**
    MPB = PBL[i];
    HSensorID = v_{H_i};
  end if
end for
return HSensorID.

is $PB(P_{(v_s,v_k)})$. The highest energy cost HEC from node v_s to all sink nodes in set V_H is calculated by:

$$HEC = \max_{v_k \in V_H}(EC(P_{(v_s,v_k)})). \tag{9}$$

We utilize HEC to calculate the normalized energy cost NEC of a path $P_{(v_s,v_j)}$ from sensor node v_s to any sink node $v_j \in V_H$:

$$NEC(P_{(v_s,v_j)}) = EC(P_{(v_s,v_j)})/HEC. \tag{10}$$

PBEC routing scheme is described as follows:

$$\begin{aligned} &\max_{v_k \in V_H}\Big(\max_{\gamma \, candidate\ paths:\, P_{(v_s,v_k)}}(PBEC_{v_s,v_k})\Big) \\ &PBEC_{v_s,v_k} = \frac{(PB(P_{(v_s,v_k)}))^{\beta}}{(NEC(P_{(v_s,v_k)}))^{\alpha}}, \end{aligned} \tag{11}$$

where γ represents the number of candidate paths from source node v_s to sink node v_k under consideration, and both α and β are exponents indicating the weight of energy cost and path bottleneck in determining the path. Note that when $\gamma = 1$, $\alpha = 1$ and $\beta = 0$, PBEC reduces to the minimum energy cost routing algorithm, and when $\gamma = 1$, $\alpha = 0$ and $\beta = 1$, PBEC reduces to the path bottleneck-oriented routing algorithm. In practical implementation, we take logarithm in Eq. 11 for simplicity. The pseudo-code for the path bottleneck-oriented and energy cost-based routing algorithm is shown in Algorithm 3.

Algorithm 3 PBEC: PB-oriented and EC-based
Input: source node v_s and its routing table (RT)
Output: the label of the selected H-Sensor

HEC = highest path energy cost;
$HEC = EC(P_{(v_s,v_{H_1})})$, $v_{H_1} \in V_H$;
for all paths $P_{(v_s,v_k)}$ from v_s to $v_k \in V_H$ in RT **do**
  if $EC(P_{(v_s,v_k)}) > HEC$ **then**
    $HEC = EC(P_{(v_s,v_k)})$;
  end if
end for
PBL = path bottleneck list;
Execute the first part of Alg. 2 to calculate PBL;
HSensorID=v_{H_1};
MaxPBEC = $\alpha \cdot (\lg(HEC) - \lg(EC(P_{(v_s,v_{H_1})}))) + \beta \cdot \lg(PBL[1])$;
for all paths $P_{(v_s,v_k)}$ in RT, $v_k \in V_H$ **do**
  temp = $\alpha \cdot (\lg(HEC) - \lg(EC(P_{(v_s,v_k)}))) + \beta \cdot \lg(PBL[k])$;
  if temp $>$ MaxPBEC **then**
    MaxPBEC = temp;
    HSensorID = k;
  end if
end for
return HSensorID.

E. A numeric example

We give a numeric example below to illustrate the above three routing algorithms. Consider the sensor network topology shown in Fig. 2. The energy of every sensor node is initialized to be 30 units and the link communication cost is represented by the number of energy units labeled over each link.

- EC: energy cost-based routing algorithm

The energy cost of the paths between source node v_6 and sink nodes v_1, v_2, and v_3 is calculated as

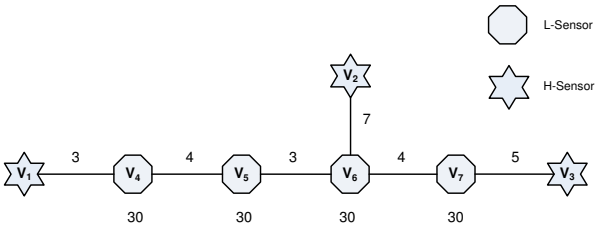


Fig. 2. Initial state of a sensor network.

follows:

$$EC(P_{(v_6,v_1)}) = LC(L_{(6,5)}) + LC(L_{(5,4)}) + LC(L_{(4,1)}) = 3+4+3 = 10,$$
$$EC(P_{(v_6,v_2)}) = 7,$$
$$EC(P_{(v_6,v_3)}) = LC(L_{(6,7)}) + LC(L_{(7,3)}) = 4+5 = 9.$$

According to Eq. 7, we choose path $P_{(v_6,v_2)}$ for data transmission.

- PB: path bottleneck-oriented routing algorithm

The energy levels or the numbers of data transmissions at nodes v_6 and v_7 on path $P_{(v_6,v_3)}$ are calculated as:

$$EL(v_6) = \lfloor \tfrac{30}{4} \rfloor = 7,$$
$$EL(v_7) = 30/5 = 6,$$

where the denominators 4 and 5 are the communication cost over links $L_{6,7}$ and $L_{7,3}$, respectively, towards the destination node v_3. Therefore, the path bottleneck, i.e. the minimum of all the energy levels on the path, $PB(P_{(v_6,v_3)}) = 6$. Similarly, we can calculate the path bottleneck $PB(P_{(v_6,v_1)}) = 7$ and $PB(P_{(v_6,v_2)}) = 4$. According to Eq. 8, we choose the path with the maximum path energy level, i.e. $P_{(v_6,v_1)}$.

- PBEC: path bottleneck-oriented and energy cost-based routing algorithm

In the EC routing algorithm, according to Eqs. 9 and 10, we obtain the highest energy cost $HEC = EC(P_{(v_6,v_1)}) = 10$ and the normalized energy cost NEC for each path. Also, in the PB routing algorithm, we obtain the path bottleneck of each path. Plugging these quantities in Eq. 11 yields the $PBEC$ value of every path. Assuming $\alpha = 1.5$ and $\beta = 1$, we have:

$$PBEC_{v_6,v_1} = \left(\tfrac{10}{10}\right)^{1.5} \times 7 = 7.00,$$
$$PBEC_{v_6,v_2} = \left(\tfrac{10}{7}\right)^{1.5} \times 4 = 6.83,$$
$$PBEC_{v_6,v_3} = \left(\tfrac{10}{9}\right)^{1.5} \times 6 = 7.03.$$

According to the PBEC routing algorithm, we select sink node v_3 for data transmission from source node v_6. PBEC combines the advantages of the EC and PB routing algorithms to balance the energy consumption of sensor nodes and prolong the network lifetime.

V. Implementation

We implemented three routing algorithms, namely minimum energy cost routing (EC), path bottleneck-oriented routing (PB), and path bottleneck-oriented and energy cost-based routing (PBEC) in Java and conducted simulations using a network of sensor nodes and sinks on a Linux 2.6.20 PC workstation equipped with dual 3.2 GHz Intel Xeon CPUs and 4 GBytes memory. The sensor nodes are distributed randomly in a planar 100×100 area and the sinks are deployed at its boundaries. Some default values of the parameters used in the experiments are tabulated in Table I.

TABLE I
Parameters and default values.

Parameter	Symbol	Value
Number of L-sensors	N	100
Number of H-sensors	n	4
Surveillance range	m×m	10×10
Influential exponent of CR	α	0~1
Influential exponent of ER	β	0~1

Currently, the lifetime of a sensor network is defined as the time for the first node [24], [6], [17], [22] or a certain percentage of network nodes [27], [9], [39] to run out of power. We adopt *cooperative lifetime* as the term for the former definition and *whole lifetime* for the latter. In the cooperative lifetime (CL) scheme, all the sensors must cooperate with each other and hence any single sensor's failure results in the breakdown of the entire network. The whole lifetime (WL) is defined as the lifetime from the startup of the network to the point when a specific percentage of sensors are not able to transmit any data.

- Experiment 1 – Cooperative lifetime of EC with different numbers of H-sensors

This experiment is designed to study the effect of the number of H-sensors on the cooperative lifetime performance of the network using EC routing protocol. Fig. 3 shows the lifetime of the network in response to different numbers of H-sensors. The horizontal axis represents the number of H-sensors and the vertical axis represents the lifetime of the network calculated in rounds.

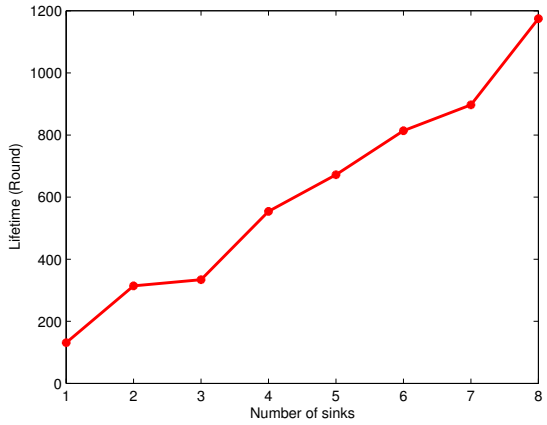


Fig. 3. Lifetime of EC routing with different numbers of sinks.

From Fig. 3, we observed that increasing the number of H-sensors significantly improves the lifetime of the network. The average distance from L-sensors to H-sensors decreases when the number of H-sensors increases, resulting in a longer network lifetime.

- Experiment 2 – Cooperative lifetime comparison of EC, PB and PBEC

This experiment is designed to compare the lifetime performance of EC, PB and PBEC. As shown in Fig. 4, the lifetime of PB is longer than that of EC and the lifetime of PBEC is longer than that of PB. The minimum energy cost path selected by EC may involve some sensors that already have very low residual energy, which accelerates the death of the first node in the network. Instead, PB and PBEC take the residual energy of sensors into consideration and avoid transmitting data through sensors with already low residual energy, resulting in a longer network lifetime.

- Experiment 3 – Whole lifetime comparison of EC, PB and PBEC

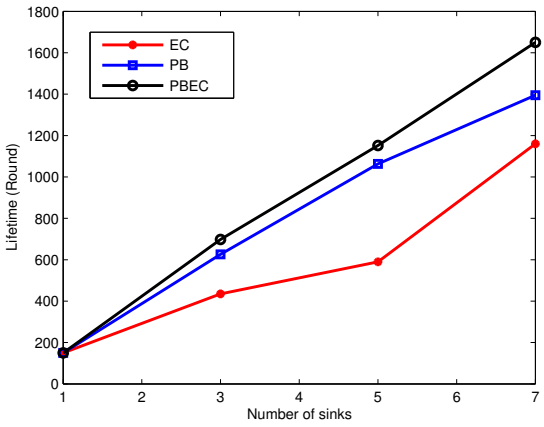


Fig. 4. Lifetime comparison of EC, PB and PBEC under CL.

This experiment is designed to compare the whole lifetime of EC, PB and PBEC with 4 H-sensors when we vary the percentages of dead sensors. In PBEC, we consider $\gamma = 2$ paths from a sensor node to each sink. From Fig. 5, we observed that PBEC achieves the longest lifetime under different percentages of dead sensor nodes, which justifies the superior lifetime performance of PBEC over EC and PB.

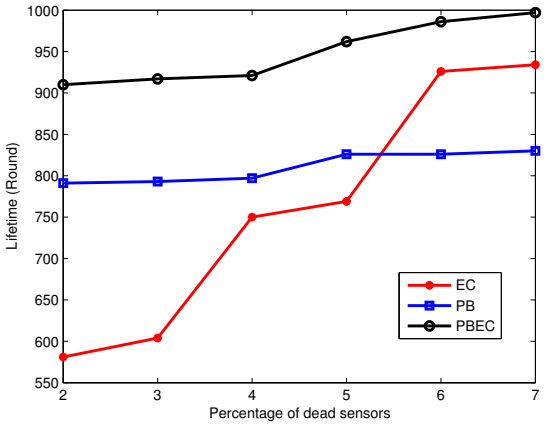


Fig. 5. Lifetime comparison of EC, PB and PBEC under WL.

- Experiment 4 – Lifetime of PBEC with different α, β values

This experiment is designed to investigate the effect of choosing different values for weight coefficients α and β on the network lifetime performance. We collected the lifetime measurements of PBEC using

different values of α and β with the constraint of $\alpha+\beta=1$. The performance curve shown in Fig. 6 exhibits a unimodal trend. In the experiments, the best values for α and β are chosen to maximize the network lifetime.

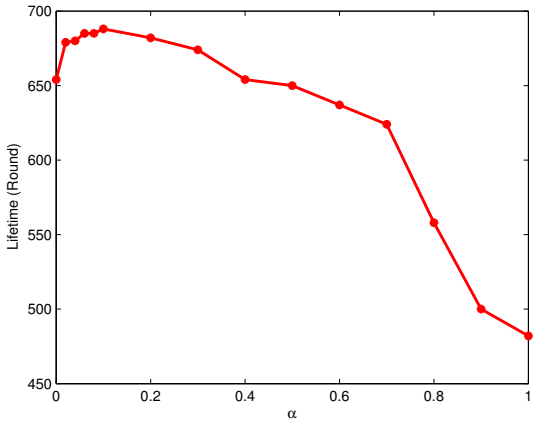


Fig. 6. Lifetime of PBEC with different α and β values.

VI. Conclusions

Single-sink sensor networks have some performance limitations mainly caused by unbalanced energy consumption of sensor nodes. The objective of this work was to investigate an innovative and robust network architecture of HSNs and design energy-efficient routing protocols for HSNs. We presented the system architecture of HSNs and proposed an energy-efficient path bottleneck-oriented and energy cost-based routing algorithm, PBEC, which targets optimizing the balance of energy consumption of sensor nodes to prolong the network lifetime. Extensive simulation results showed that the proposed PBEC algorithm significantly improves the lifetime of sensor networks compared to other algorithms.

It is our future interest to further investigate the performance effect of the weight exponents in the current form of PBEC and design new methods to optimize the multi-objective problems in HSNs. We will also look into the performance effect of the number of candidate paths in PBEC path selection. One important issue is to balance the tradeoff of routing quality and computation overhead resulted from using multiple paths.

Acknowledgments

This research is sponsored by National Science Foundation under Grant No. CNS-0721980 and Oak Ridge National Laboratory, U.S. Department of Energy, under Contract No. PO 4000056349 with University of Memphis.

References

[1] Sensor Deployment in the James Reserve. http://dms.jamesreserve.edu.

[2] Babel, a loop-free distance-vector routing protocol. http://www.pps.jussieu.fr/ jch/software/babel.

[3] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communication Magazine*, 40(8):102–114, 2002.

[4] The Embedded Networks Lab at USC.

[5] D. Braginsky and D. Estrin. Rumor routing algorithm for sensor networks. In *First Workshop on Sensor Networks and Applications*, 2002.

[6] J.H. Chang and L. Tassiulas. Maximum lifetime routing in wireless sensor networks. *IEEE/ACM Transactions on Networking*, 12(4):609–619, 2004.

[7] R. Cristescu and B. Beferull-Lozano. Lossy network correlated data gathering with high-resolution coding. In *Proc. of IEEE IPSN*, 2005.

[8] A. Das and D. Dutta. Data acquisition in multiple sink sensor networks. In *Proc. 2nd International Conference on Embedded Sensor Systems*, 2004.

[9] E. Duarte-Melo and M. Liu. Analysis of energy consumption and lifetime of heterogeneous wireless sensor networks. In *GLOBECOM 02*, volume 1, page 2125.

[10] E.J. Duarte-Melo and M. Liu. Data-gathering wireless sensor networks: organization and capacity. *Computer Networks (COMNET) Special Issue on Wireless Sensor Networks*, 43(4):519–537, Nov. 2003.

[11] S.D. Feller. Tracking and imaging humans on heterogeneous infrared sensor arrays for law enforcement applications. In *SPIE Aerosense*, April 2002.

[12] The Center for Embedded Networked Sensing (CENS) at UCLA.

[13] L. Girod, J. Elson, A. Cerpa, T. Stathopoulos, N. Ramanathan, and D. Estrin. Emstar: a software environment for developing and deploying wireless sensor networks. In *Proc. of USENIX 04*, 2004.

[14] L. Girod, T. Stathopoulos, and N. Ramanathan. A system for simulation, emulation, and deployment of heterogeneous sensor networks. In *Proc. of the 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Nov. 2004.

[15] R. Govindan, E. Kohler, and D. Estrin. Tenet: An architecture for tiered embedded networks. Technical report, CENS Technical Report, Nov. 2005.

[16] P. Gupta and P.R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, IT-46(2):388–404, March 2000.

[17] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocols for wireless microsensor networks. In *Proc. of Hawaiians Int'l Conf. on Systems Science*, January 2000.

[18] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proc. of ACM MOBICOM'00*, Boston, MA, Aug. 2000.

[19] S.S. Iyengar, D.N. Jayasimha, D. Nadig, and D.K. Pradhan. A versatile architecture for the distributed sensor integration problem. *IEEE Transactions on Computers*, 43(2), Feb. 1994.

[20] D.N. Jayasimha and S.S. Iyengar. Information integration and synchronization in distributed sensor networks. *IEEE Trans. Systems, Man, and Cybernetics*, 21(5):1032–1043, Sept./Oct. 1991.

[21] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Pehand, and D. Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *Proceedings of the Tenth ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, October 2002.

[22] K. Kalpakis, K. Dasgupta, and P. Namjoshi. Maximum lifetime data gathering and aggregation in wireless sensor networks. In *Proceedings of the 2002 IEEE International Conference on Networking*, pages 685–696, Atlanta, Georgia, August 2002.

[23] H. Kim, Y. Seok, N. Choi, Y. Choi, and T. Kwon. Optimal multi-sink positioning and energy-efficient routing in wireless sensor networks. *Lecture Notes in Computer Science*, 3391:264 – 274, Jan. 2005.

[24] S. Madden, M.J. Franklin, J.M. Hellerstein, and W. Hong. Tag: A tiny aggregation service for ad-hoc sensor networks. In *OSDI*, 2002.

[25] A. Mainwaring, J. Polastre, R. Szewczyk, and D. Culler. Wireless sensor networks for habitat monitoring. In *ACM Workshop on Sensor Networks and Applications*, 2002.

[26] E.I. Oyman. *Multiple Sink Location Problem and Energy Efficiency in Large Scale Wireless Sensor Networks*. PhD thesis, Bogazici University, 2002.

[27] E.I. Oyman and C. Ersoy. Multiple sink network design problem in large scale wireless sensor networks. In *Proc. International Conferences on Communication*, June 2004.

[28] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC3561, July 2003.

[29] C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. In *Proc. of SIGCOMM*, 1994.

[30] A. Perrig. Spins: Security protocols for sensor networks. In *Proc. of MOBICOM*, 2001.

[31] L. Prasad, S.S. Iyengar, R.L. Kashyap, and R.N. Madan. Functional characterization of sensor integration in distributed sensor networks. *IEEE Transactions on SMC*, Sept. 1991.

[32] S. Rhee, D. Seetharam, and S. Liu. Techniques for minimizing power consumption in low data-rate wireless sensor networks. In *Proc. of IEEE Wireless Communications and Networking Conference (WCNC04)*, Atlanta, GA, March 2004.

[33] E.M. Royer. A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communication*, April 1999.

[34] L. Schwiebert, S.K.S. Gupta, and J. Weinmann. Research challenges in wireless networks of biomedical sensors. In *Proc. ACM/IEEE MOBICOM '01*, pages 151 –165, 2001.

[35] D. Steere, A. Baptista, D. McNamee, C. Pu, and J. Walpole. Research challenges in environmental observation and forecasting systems. In *Proc. ACM/IEEE MOBICOM '00*, Boston, August 2000.

[36] R. Szewczyk, J. Polastre, A. Mainwaring, J. Anderson, and D. Culler. An analysis of a large scale habitat monitoring application. In *Proc. of the 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Nov. 2004.

[37] H. Wang, D. Estrin, and L. Girod. Preprocessing in a tiered sensor network for habitat monitoring. In *Proc. of IEEE Conf. on Acoustics, Speech, and Signal Processing*, Hong Kong, China, April 2003.

[38] R. Wesson. Network structures for distributed situation assessment. *IEEE Transactions on Systems, Man and Cybernetics*, pages 5–23, Jan. 1981.

[39] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. In *Proc. ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, page 7084, Rome, Italy, July 2001.

[40] Y. Xue, Y. Cui, and K. Nahrstedt. Maximizing lifetime for data aggregation in wireless sensor networks. *Mobile Networks and Applications*, 10(6):853 – 864, Dec. 2005.

[41] M. Yarvis and N. Kushalnagar H. Singh. Exploiting heterogeneity in sensor networks. In *Proc. of IEEE INFOCOM*, Miami, FL, March 2005.

Contour Guided Dissemination in 4-Neighbor Wireless Mesh Topologies

Kranthi K. Mamidisetty and Shivakumar Sastry

***Abstract*—A mesh topology is one in which each node communicates directly with a set of neighboring nodes. Using regular topologies in which each node has four neighbors, we characterize the structure of the union of all the shortest paths between a pair of nodes. Further, the multiple shortest paths between a pair of nodes cannot be exploited by using a simple traffic spreading strategy. When a source sends M messages to the sink, and every node spreads the messages over every available shortest path uniformly, we show that nodes along one shortest path will always handle more messages than the nodes along other shortest paths. We then present rules that ensure that nodes, which are at a same distance from the source, along all the paths handle roughly the same number of messages. Finally, we present simulation results that validate the analytical conclusions. In the future, similar methods for dissemination can be developed for general topologies.**

I. Introduction

Recent advances in integrated computation and communication platforms [8] offer tremendous opportunities for future engineered systems. Sensor networks and networked sensor-actuator systems [7] are two examples of emerging systems. In many engineered systems, the nodes are located at well-designed, fixed, locations. To ensure that the system is robust, multiple paths for communication between a pair of nodes are designed into the system. This paper addresses some of the challenges in disseminating messages by exploiting all the available paths between a pair of nodes.

In addition to the message overheads, routing protocols used in wired networks are not suitable for emerging systems. Because the nodes are resource-constrained, it is not desirable to maintain routing tables. When there are multiple monitoring stations, nodes must maintain a different routing table for each monitoring station. Routing methods that are based on shortest path algorithms select a single path from a node to a sink. Because all the messages propagate along this path, the nodes along this path handle more messages than the other nodes resulting in degraded Quality of Service (QoS).

Contour guided dissemination [3] is a recent method for multipath dissemination. A *Contour* is the union of all the shortest paths between a pair of nodes. We present the structure of contours in a regular mesh topology when each node has three neighbors. In Section III we briefly review contour guided dissemination. We then show that to benefit from the multiple paths in a contour, it is not sufficient to uniformly spread the messages among the available paths; surprisingly, nodes along one path in the contour will always handle more messages than other nodes. Optimal rules for spreading messages are presented. We present simulation results in Section VI and conclusions in Section VII.

II. Background

The problem of disseminating messages in networked embedded systems is challenging because of the resource constraints at the node-level, limited bandwidth, high bit-error rates and irregular propagation of the wireless links [8], [16]. Because the wireless medium has a relatively low bandwidth, it is not desirable for every node to communicate with every other node directly. The immense scale makes it infeasible to associate a network address with each node in a networked embedded system. Propagation of messages between non-neighboring nodes, without relying on unique network identifiers for each node, is referred to as dissemination in the sensor network literature [6], [9], [10]. The dissemination method we describe in this paper is similar to a gradient dissemination scheme [1]; the cost metric is derived from the structure of the contour we discuss in Section III.

Multipath routing has been the focus of a number of investigations with fault tolerance, higher aggregate bandwidth, and load balancing as some of the motivating factors [13]. The QoS aspects of multipath routing have been addressed in [4]. The split multipath routing protocol maintains maximally edge disjoint paths [11]. Braided multi-paths discussed in [5] are useful when the routing is coupled with diagnostic or prognostic methods to select alternative paths. In-

{km50,ssastry}@uakron.edu

stead of focusing on route discovery and maintenance, our focus in this paper is on precisely characterizing all the shortest multiple paths between a pair of nodes in a mesh topology and exploiting these paths to improve the QoS.

We consider mesh topologies that are obtained by embedding a given set of nodes, $\mathcal{N}$, on a 2D-BaseGrid. Each location on 2D-BaseGrid is identified by a unique ordered pair (i, j). The distance between two consecutive locations on the grid, i.e., between (i, j) and $(i, j + 1)$ or between (i, j) and $(i + 1, j)$, is b.

An embedding function, Ξ, assigns a location on 2D-BaseGrid and a transmission range to each node, $n_i \in \mathcal{N}$.

$$\Xi : n_i \in \mathcal{N} \rightarrow (\mathbb{N} \times \mathbb{N} \times \mathbb{R}). \quad (1)$$

We assume that all nodes within the transmission range of a source node reliably receive messages sent by the node. Following [3], we use the notation Ξ_q to represent an embedding function in which each node has q neighbors. Different mesh topologies are obtained by specifying different embedding functions. Suppose that the nodes in $\mathcal{N}$ must be embedded along R rows and C columns, starting at location (X, Y). Let Ξ_q denote the embedding in which each node, $n \in \mathcal{N}$, communicates directly with q neighbors. For $q = 4$, the embedding function

$$\Xi_4 : n_i \in \mathcal{N} \rightarrow (X+(i-1) \bmod C, Y+\lfloor \frac{i-1}{C} \rfloor, b) \quad (2)$$

yields the mesh topology shown in Figure II, where $|\mathcal{N}| = 16$, $X = Y = 1$. Since the transmission range for each node is b, it can only communicate with its four immediate neighbors.

III. Structure of Contours in Ξ_4

Consider a pair of nodes $n_{q,r}$ and $n_{i,j}$ in a Ξ_4 embedding [1] as shown in Figure 2. Any path from $n_{q,r}$ towards $n_{i,j}$ that reduces the distance between the nodes either along the x axis or the y axis in each step is a shortest path. In this figure, all the nodes that appear on a shortest path are shown in dark color and the nodes in the contour shaded dark.

Definition 1: A contour is the union of all the shortest paths between a pair of nodes.

[1] We consistently use $n_{q,r}$ to represent the source and $n_{i,j}$ to represent the sink in a contour.

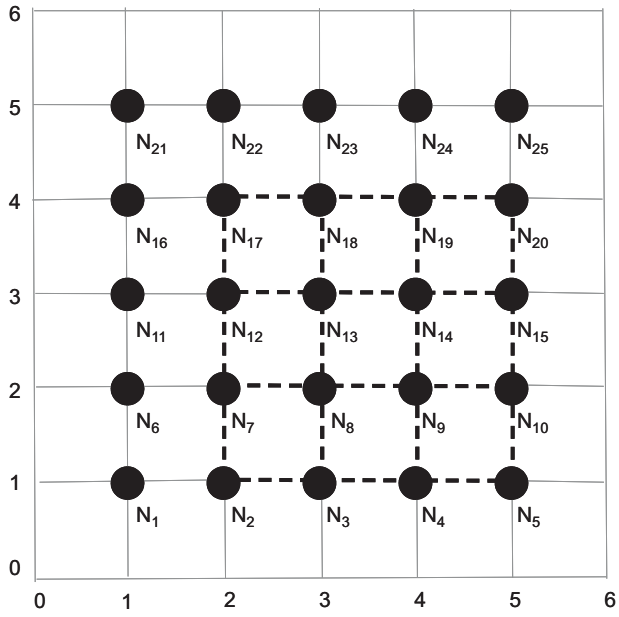


Fig. 1. Embedding for Ξ_4

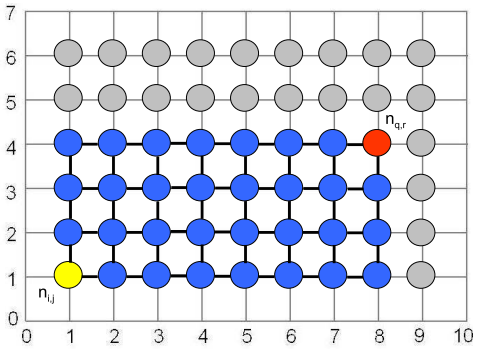


Fig. 2. An Example Contour - the union of all shortest paths between a pair of nodes.

To characterize the structure of the contour, we define the following quantities:

$$\Delta_x(n_{q,r}, n_{i,j}) = |q - i| \quad (3)$$

$$\Delta_y(n_{q,r}, n_{i,j}) = |r - j| \quad (4)$$

$$\Delta_{xy}(n_{q,r}, n_{i,j}) = |\Delta_x(n_{q,r}, n_{i,j}) - \Delta_y(n_{q,r}, n_{i,j})| \quad (5)$$

When the context is clear, we use Δ_x, Δ_y and Δ_{xy} without specifying the source and the sink. For example, in Figure 2 $\Delta_x = 7$, $\Delta_y = 3$ and $\Delta_{xy} = 4$.

A. Length of Shortest Paths

In the Ξ_4 embedding, the length of the shortest path, d, between two nodes $n_{q,r}$ and $n_{i,j}$ is

$$d = \Delta_x + \Delta_y \tag{6}$$

For example, the length of the shortest path in the contour shown in Figure 2 is $\Delta_x + \Delta_y = 10$. If $n_{q,r}$ were located at $(9, 5)$, then the length of the shortest path is 12. From the well-known principle of optimality [2], it is easily seen that a node, $n_{s,t}$, is in the contour of $n_{i,j}$ and $n_{q,r}$ if and only if

$$d(n_{i,j}, n_{s,t}) + d(n_{s,t}, n_{q,r}) = d(n_{i,j}, n_{q,r}). \tag{7}$$

We now present results to characterize the contour between two nodes $n_{q,r}$ and $n_{i,j}$. **To keep the notation simple, we consider the case when $q > i$, $r > j$ and, whenever $\Delta_{xy} > 0$, $q - i > r - j$ i.e., $\Delta_x > \Delta_y$. All other cases can be handled in a similar manner**.

B. Contour Structure

We first note from Figure 2 that in the embedding Ξ_4, a node $n_{s,t}$ communicates with four of its neighbors, but only two of these nodes, will lie along the shortest path from $n_{s,t}$ to the sink satisfying the condition in Equation (7). For the corner nodes only one its four neighbors will satisfy the condition in Equation (7).

Lemma 1: In the Ξ_4 embedding, the contour of $n_{q,r}$ and $n_{i,j}$ will comprise a single shortest path iff $\Delta_y = 0$.

Proof: Given $\Delta_y = 0$, the length of the shortest path is $\Delta_x = q - i$. When $\Delta_y = 0$, the path $P = < n_{q,j}, n_{q-1,j} \cdots n_{i+1,j}, n_{i,j} >$ is a shortest path and all nodes that lie along P satisfy Equation (7). So this is one shortest path. If there is an additional shortest path then there should be some other node which is a neighbor of the nodes on path P and which also satisfies Equation (7). Any node $n_{s,t}$ in P has only one neighbor it can communicate with $n_{s-1,t}$, which happens to be the node in the path P. There are no other neighbors of any node on the path P which satisfy the Equation 7. Hence, there is only one shortest path if $\Delta_y = 0$.

Single shortest path in the contour means that a node $n_{s,t}$ which is on the shortest path can communicate only to one of its four neighbors, if there were more than one neighbor a node can communicate and still be in accordance with Equation (7) then it would result in more than one shortest path. So the corresponding nodes could be $n_{s-1,t}$ or $n_{s,t-1}$ as the path length should decrease as we move along the shortest path. But considering only the cases where $\Delta_x > \Delta_y$, only the node $n_{s-1,t}$ is in the contour. This is valid for any node on the shortest path we see that the y-coordinate doesn't change with the nodes there by leading to $\Delta_y = 0$.

■

Lemma 2: In Ξ_4, the contour of $n_{i,j}$ and $n_{q,r}$ comprises two edge disjoint shortest paths that are enclosed by the nodes $n_{i,j}, n_{i,j+1}, n_{q,r-1}$ and $n_{q,r}$ if and only if $\Delta_y = 1$.

Proof: Given the nodes as stated, we can use the distance criterion in Equation (7), to verify that node $n_{i,j+1}$ and node $n_{q,r-1}$ are on two different shortest paths from $n_{i,j}$ to $n_{q,r}$. If they were on a single shortest path then Equation (7) would not satsify. It now suffices to note that node $n_{i,j+2}$ and $n_{q,r-2}$ do not satisfy the distance criterion in Equation (7) and hence these are the only two shortest paths possible. It is sufficient to note that there are no neighbors for nodes on these two shortest paths which are on any shortest path following Equation (7).

Given two shortest paths between $n_{i,j}$ and $n_{q,r}$, we now show that the paths are enclosed by the four nodes as stated. We know that a path from $n_{q,r}$ to $n_{i,j}$ is a shortest path only if $\Delta_x + \Delta_y$ is reduced along each step along the path. In the Ξ_4 embedding, there are only two neighbors of $n_{q,r}$ with a lower Δ_x or Δ_y , namely $n_{q-1,r}$ and $n_{q,r-1}$. $n_{q+1,r}$ and $n_{q,r+1}$ does not satisfy Equation (7). Using similar arguments in the neighborhood of $n_{i,j}$, we conclude that only node $n_{i,j+1}$ and $n_{i+1,j}$ can be on any shortest path between $n_{i,j}$ and $n_{q,r}$. The nodes $n_{i+1,j}$ and $n_{q,r-1}$ are on a same shortest path similarly the other two nodes $n_{i,j+1}$ and $n_{q-1,r}$ are on a different shortest path. We can see that there is only one neighbor each for $n_{i,j+1}$ and $n_{q,r-1}$ which satisy Equation (7). There by these two nodes act like the corner nodes bounding the two shortest paths. Hence the four nodes enclose the contour as stated.

■

Theorem 1: In Ξ_4, the contour of $n_{i,j}$ and $n_{q,r}$ is a rectangle bounded by the nodes $n_{i,j}$, $n_{i,j+\Delta_y}$, $n_{q,r-\Delta_y}$ and $n_{q,r}$ for $\Delta_{xy} > 0$.

Proof: Mathematical Induction principle will be used to prove this theorem. The theorem states that corner nodes of the contour are depended on Δ_y.For $\Delta_y = 1$ the corner nodes as stated by the theorem are $n_{i,j}$, $n_{i,j+1}$, $n_{q,r-1=j}$ and $n_{q,r}$ which are same derived

from lemma2. The theorem holds true for $\Delta_y = 1$. Assume by induction hypothesis that the result is true for $\Delta_y = u$ and u $\leq \Delta_x$. To prove that the result hold true for $\Delta_y = u + 1$. The corner nodes of the contour for $\Delta_y = u$ are $n_{i,j+u}$ and $n_{q,r-u}$.

Consider the contour which has source at $n_{q,r+1}$ and sink at $n_{i,j+u}$ but $j + \Delta_y = r$. So the sink is at $n_{i,r}$, these nodes form a contour with $\Delta_y = 1$ so the corresponding corner nodes for the contour from the theorem are $n_{i,r}$, $n_{i,r+1}$, $n_{q,r+1-1}$ and $n_{q,r+1}$.

The above observation along with induction hypothesis derive that the contour of $n_{i,j}$ and $n_{q,r+1}$ is bounded by the nodes $n_{i,j+(r+1-j)} = n_{i,r+1}$ and $n_{q,r+1-(r+1-j)} = n_{q,j}$ which are the same as stated by the theorem. ■

A similar characterization of the structure of contours in the Ξ_8 and Ξ_3 embeddings are reported in [3].

Given the relative locations of the source and sink, $n_{q,r}$ and $n_{i,j}$, by the results in this section, the structure of the contour (e.g., location of corner nodes which can be computed using Δ_{xy} and Δ_y) is known. Thus, each node can compute its own location in the contour and use the structure of the contour to determine the number and location of its upstream and downstream neighbors. In Section V, we show that this location information is necessary to effectively exploit all the available shortest paths in a contour in the Contour Guided Dissemination proposed in this paper.

IV. Effects of Uniform Spreading

Uniform spreading is a straight forward strategy to spread the messages in a contour. The source, and each intermediate node along every path, will spread successive messages in a round-robin fashion over the available paths. For example, in Figure 3, the source, $n_{q,r}$, sends 100 messages to two of its neighbors which are on the shortest paths. This neighbors spreads the 50 messages each over its two neighbors. These nodes in turn send messages to their neighbors until all the messages arrive at the sink, $n_{i,j}$.

Definition 2: A *row* in a contour is a set of nodes that are at the same distance (Equation (6)) from the source.

There are 9 rows, excluding the source and the sink, in the contour shown in Figure 3. The number of nodes in every row increases by 1 from the number of nodes in its preceding row in *Expansion* the same number of nodes in the *Propagation* and number of nodes decreases by 1 from its preceding row

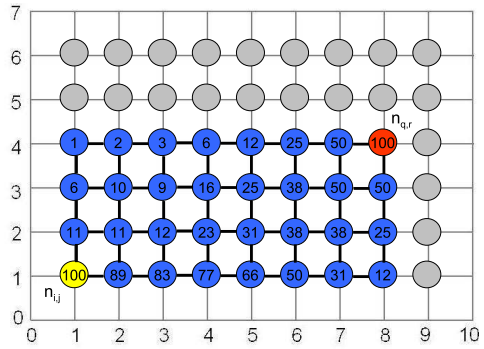


Fig. 3. When all the nodes spread messages uniformly, one of the paths in a Ξ_4 contour will handle more messages.

in *Contraction* regions. Further, notice that even though every node spreads the messages it handles over all the available paths, many nodes closer to the sink handle more messages than other nodes. In this section, we characterize this phenomenon precisely.

Observation 1: Note that for any node $n_{s,t}$ in an Ξ_4 contour, its neighbors $n_{s,t+1}$ or $n_{s+1,t}$ cannot be on a shortest path from $n_{s,t}$ to $n_{i,j}$.

Proof: The shortest path from a node $n_{s,t}$ to $n_{i,j}$ in Ξ_4 is of the length measured by $\Delta_x + \Delta_y$. So every step along the shortest path towards the node $n_{i,j}$ should reduce the length by 1 from $n_{s,t}$ which is possible only by the nodes labelled $n_{s-1,t}$ and $n_{s,t-1}$. The distance measured from $n_{s,t+1}$ or $n_{s+1,t}$ increases by 1 from $n_{s,t}$. Which doesn't go with (Equation (6)). ■

When a message travels along a shortest path to the sink, the distance to the sink reduces along each step of the shortest path by one. In a contour, a node $n_{s,t}$ may be able to send the messages it handles to at most four of its neighbors, i.e., $n_{s,t-1}$, $n_{s,t+1}$, $n_{s-1,t}$, and $n_{s+1,t}$.

A. Loading under Uniform Spreading in Ξ_4

In the embedding Ξ_4, notice from Figure 3 that under uniform spreading, all the nodes $n_{s,t}$ in layers $1, 2, \cdots, \Delta_y$ of the contour either send their messages to the corresponding, unique, neighbor on the next row or divide the messages they handle into two parts and send each part along one of their two neighbors that are closer to the sink. Thus, we can conclude that the ratio of the messages handled by a node in a row, to the smallest number of messages handled

by another node in the same row, follow the binomial coefficients[2].

Lemma 3: Given a Ξ_4 contour with $\Delta_y(n_{q,r}, n_{i,j}) = 1$, if $n_{q,r}$ sends M messages to $n_{i,j}$, nodes along one path handle $M-1$ messages after $\log_2 M$ steps along the shortest path from $n_{q,r}$ to $n_{i,j}$.

Proof: For $\Delta_y = 1$ the contour will comprise of only two layer shortest paths one with all the nodes having $\Delta_y = 0$ and $\Delta_y = 1$. The length of the shortest path is $\Delta_x + \Delta_y = \Delta_x + 1$, we can see that atmost we can move Δ_y steps in the vertical axis and Δ_x steps in the horizontal axis to reach the *Sink* from *Source*. A node in the contour can only communicate with nodes as seen in the 1. The messages that are handled by nodes with $\Delta_y = 1$ half of them are sent to nodes with $\Delta_y = 0$. We can easily see that this follows from the binomial expansion and after $\log_2 M$ steps of communication the nodes on $\Delta_y = 0$ will handle $M-1$ messages. Because the nodes with $\Delta_y = 0$ cannot communicate further down and could only communicate with nodes which decrease the corresponding Δ_x to satisfy (Equation (6)).

■

To show that uniform spreading results in loading one path of the contour when $\Delta_{xy} > 0$, let us consider subpaths in a contour labeled as shown in Figure 4. All the nodes on Path-k have $\Delta_y = k$ w.r.t. the sink, $n_{i,j}$.

Lemma 4: Under uniform spreading in a contour in Ξ_4, in every row, nodes along the subpath with $\Delta_y = 0$ w.r.t the sink, will always handle more messages than the nodes along other paths.

Proof: The length of the shortest path from source $n_{q,r}$ to sink $n_{i,j}$ is $\Delta_x + \Delta_y$. Any node $n_{s,t}$ in the contour on layer $\Delta_y = k$ can communicate $\Delta_y - k$ steps in vertical direction after which it faces condition like it could communicate only with nodes which have $\Delta_y = 0$ and communicating with nodes which decrease only Δ_x to satisfy (Equation (6)) and the shortest path length.

By the communication patterns in the contour a node in the some $\Delta_y = k$ can communicate only with nodes which are its neighbors and which have $\Delta_y = k$ or $\Delta_y = k-1$ to satisfy (Equation (6). So any node $n_{s,t}$ in the contour which communicates to its neighbors will reach the line $\Delta_y = 0$ before $\Delta_x = 0$ as $\Delta_y < \Delta_x$. So the nodes with $\Delta_y = 0$ cannot communicate along the lines which decrease Δ_y so they communicate along the nodes which have Δ_x reducing . So all the messages start reaching the nodes with $\Delta_y = 0$ there by making the nodes along $\Delta_y = 0$ handle more messages than $\Delta_y = k$ for $k > 0$. ■

We note that a node in a lower numbered subpath (layer) cannot forward messages to a node in a higher numbered path without increasing the length of the shortest path.

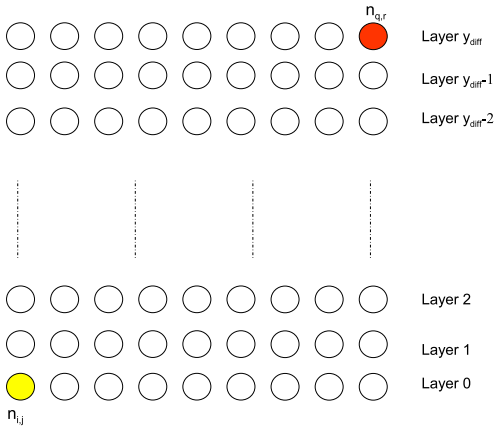


Fig. 4. Layers in Ξ_4.

V. Optimal Spreading

We use the structure of contours to derive rules that ensure that the nodes in a given row of a contour handle roughly the same number of messages. When the traffic is spread using these rules, the effects of loading caused under uniform spreading are mitigated.

Some nodes in the contour handle more messages than the other nodes because of the structure of the shortest paths. In particular, loading occurs because nodes along certain paths can only send messages to neighboring nodes that have lower Δ_y values with respect to the sink. Consider the contour shown in Figure 5. This view shows three regions that are called *Expansion, Propagation* and *Contraction* regions. In the expansion region, nodes spread the messages handled over the available paths using rules presented in this section. In the propagation region, nodes do not spread messages; however, the messages are propagated along one shortest path. In the contraction region, nodes coalesce messages from multiple nodes in the preceding rows. Loading occurs when nodes in the propagation region spread the messages handled instead of propagating the messages along one

[2] This is true to the extent that the number of messages can be equally divided among the nodes in a row.

shortest path.

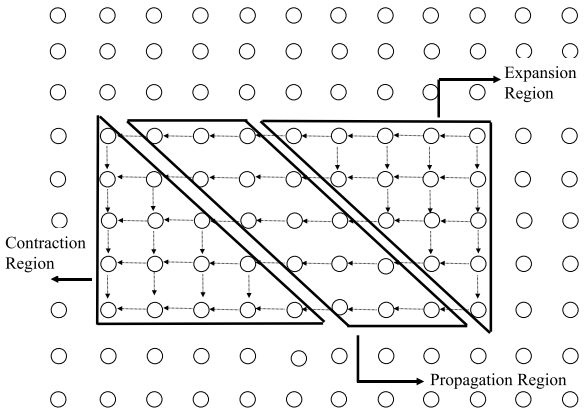


Fig. 5. Expansion, Propagation and Contraction Regions in an Ξ_3 Contour.

Let w be the number of nodes in some row of the expansion region of a contour in Ξ_3. The rules for optimal spreading are based on the following labeling of the nodes in every row of a contour. The nodes on both sides of the middle node are labeled as $1, 2, 3, \cdots, \lfloor w/2 \rfloor$. For example, Figure 6 illustrates such a labeling for two rows in an expansion region with w and $w+1$ nodes. These labels can be easily computed by each node after computing its own location based on the structure of the contour presented in Section III [3].

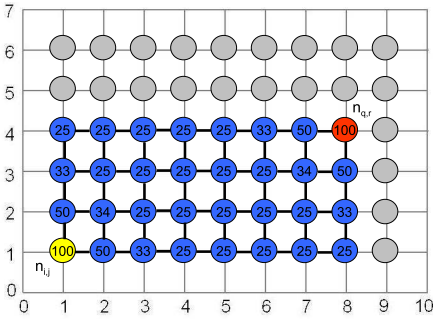


Fig. 6. Spreading in the Expansion Region

Notice that in the expansion region, the number of nodes in the suceeding row remains the same in alternating rows. To specify rules for optimal spreading, we must consider two scenarios depending on the parity of w:

Theorem 2: For Optimal dissemination in the expansion region of a contour in Ξ_3,

1. When w is even,
   (a) Node n_p^m sends $\frac{M}{w} \times \frac{w+1-p}{w+1}$ messages to the node n_p^{m+1} and $\frac{M}{w} \times \frac{p}{w+1}$ messages to the node n_{p+1}^{m+1}.
   (b) The middle node, $n_{w/2}^m$, sends $\frac{M}{w} \times \frac{w/2+1}{w+1}$messages to $n_{\lfloor (w+1)/2 \rfloor}^{m+1}$ and $\frac{M}{w} \times \frac{w/2}{w+1}$ messages to the node $n_{\lceil (w+1)/2 \rceil}^{m+1}$.
2. When w is odd,
   (a) Node n_p^m sends $\frac{M}{w} \times \frac{w+1-p}{w+1}$ messages to node n_p^{m+1} and $\frac{M}{w} \times \frac{p}{w+1}$ messages to the node n_{p+1}^{m+1}.
   (b) The middle node, $n_{\lfloor w/2 \rfloor+1}^m$, sends $\frac{M}{2w}$ messages to $n_{\lfloor (w+1)/2 \rfloor}^{m+1}$.

Proof: Given the number of messages to be transmitted from source to sink is M. Accordingly if number of nodes in a row of contour are w then the ideal number of messages to that each node should handle is $\frac{M}{w}$.

The spreading rules stated in the theorem would make sure that optimal number of messages are handled by every node in row of contour. Proving the above rules by Mathematical Induction. Lets first analyse the Odd to Even transition i.e., when w is odd. When w=1, then the node it self is the middle node and the next row has width $w+1$=2 which is even and both of them will be middle nodes for the corresponding row and the labels would be 1 and 1 for each and accordingly from the theorem the number of messages transmitted from middle node to nodes is next row is $\frac{M}{2w}$ which is $\frac{M}{2}$ equivalent to the number of messages that should be ideally handled by row with width $w+1$=2. So the case of w=1 is true.

Verifying the integrity of the rules for w=3, this time we have a single middle node and two other nodes in the row. The next row will have $w+1$=4 nodes with two middle nodes and they are labelled 2 and 2 and other nodes as 1 and 1 this follows from the labelling strategy we followed described earlier. Consider the corner node of the preceding row its lebelled 1 and it had its source of messages only from node labelled 1 in the previous row. So applying the transition n_p^m to n_p^{m+1} we have the number of messages to be $\frac{M}{3} \times \frac{3+1-1}{3+1} = \frac{M}{4}$ which the ideal number of messages to be handled by nodes in the row of width 4.

The node labelled 2 in the preceding row will get its

[3] n_p represents the node that is labeled p as illustrated in Figure 6. n_p^m represent the node that is labeled p in the m^{th} row of the contour.

input messages from nodes labelled 1 and the middle node in the previous row. so the transition n_p^m to n_{p+1}^{m+1} then it contributes for $\frac{M}{3} \times \frac{1}{4} = \frac{M}{12}$ and the middle node transmits about $\frac{M}{2*3} = \frac{M}{6}$ so the total number of messages turn out to be $\frac{M}{6} + \frac{M}{12} = \frac{M}{4}$ which again is the ideal number of messages to be handled by the nodes in a row of width 4. Similar is the case for the other two symmetrical node on other side of middle node.

Hypothesis : Let the above rules be valid for some w which is odd and each node in the corresponding row will handle $\frac{M}{w}$ number of messages. We have $\lfloor \frac{w}{2} \rfloor$ number of nodes on either side of the middle node. Consider the corner node lebelled 1 in the next row, this node has only one input source from the node labelled 1 in the previous row. Certainly the ideal number of messages to be handled by that node is $\frac{M}{w+1}$, which is same as replacing p with 1 in the transition n_p^m to n_p^{m+1}. So all the corner nodes of the row follow this rule. Now consider any other node which is not a corner node and not middle node

so n_{p+1}^{m+1} will get its messages from n_p^m and n_{p+1}^m so applying the spread rules for these trnasitions and summing them up then numbber of messages handled by n_{p+1}^{m+1} will be $\frac{M}{w} \times \frac{i}{w+1} + \frac{M}{w} \times \frac{(w+1)-(i+1)}{w+1} = \frac{M}{w+1}$

Which is ideal number of messages to be handled by nodes in a row of width $w+1$. The last remaining nodes are those which receive their share of messages from the middle node, node labelled $\lfloor \frac{w+1}{2} \rfloor$ is one such node. The source for this node messages is middle node and $\lfloor \frac{w}{2} \rfloor$ node which is equal to $\frac{M}{2w} + \frac{M}{w} \times \frac{w+1-\lfloor \frac{w}{2} \rfloor}{w+1}$ solving which will give us $\frac{M}{w+1}$ which is the ideal number of messages for nodes in a row of width $w+1$.

Same is the argument for the nodes on other side of the middle node. Similar arguments can be applied to the row transition form Even number of nodes to Odd number of Nodes. ■

A proof based on mathematical induction is presented in [12]. It is easy to see that if the number of messages handled by the nodes in a row at the beginning of the propagation region are roughly the same, this number will not change since the nodes do not spread messages. The proof for the rules in the contraction region also follow from induction.

VI. Results

To evaluate the QoS achieved by CGD in a 4-Neighbor and 6-Neighbor mesh topologies, we used a discrete event simulation that was designed and implemented in the OMNet++ framework [15].

A. QoS Metrics

In each simulation run, we counted the number of messages handled by each node. For each message, the time at which it was sent from the source and the time at which it arrived at the sink were recorded. The difference between these two times is the end-to-end latency for the message. We computed the average latency over all the messages received at the sink. In addition, we computed the Jitter, which was defined as the standard deviation of the end-to-end latencies. Message Loss rate was computed as the ratio of the number of messages that were not received at the sink to the number of messages that were sent from the source. To evaluate the effects of congestion on the performance, we considered the number of messages lost in a node due to buffer overflow. Finally, to evaluate the scalability of CGD, we used throughput, which was defined as the number of messages arriving at the sink per unit time, as a metric.

B. Simulation Approach

The multihop, multipath, propagation of messages from the source to the sink was modeled as a multistage queuing network [14]. The service time of each queue, was exponentially distributed with a mean time of 25 ms. This time represents the time required to propagate a message from the preceding node, the time required to receive a message in the node, and the time required to forward the message towards the sink. Using the channel modeling capability in OMNet++, the data rate (bandwidth) was set to 38.4 Kbps. Propagation delay and bit error rates were assumed to be zero, which are the default values in OMNet++. All the links between every pair of nodes the wireless mesh topology used the same channel model. We designed a simple message generator to serve as the source. The inter-arrival time of the messages at the source was exponentially distributed with a mean time of 15 ms.

To ensure that the simulations reflected real-world constraints on propagation and topology, we weakened the regular topologies considered for the analysis by varying the probability of success for each link in the contour between 0 (failure) and 1 (success). Before sending a message to a node on the downstream row in the contour, a random number was drawn from a uniform distribution; the message was sent only if this random number was larger than the link failure

threshold.

We used the QoS metrics described, as a basis to compare the performance of CGD with the performance achieved using single shortest path (Unipath), which is typically used in traditional distance vector routing schemes and the shortest path tree that was computed based on the structure of all the shortest paths. This shortest path was pre-computed.Because of the transient nature of link failures, we did not make any changes to the shortest path to account for link failures. This comparison was used as a watermark for acceptable performance.

The results reported here are based on a scenario in which the source sends 5000 messages, each 36 bytes long. To study the effects of queueing delays, we considered a Ξ_4 and Ξ_6 embedding with $\Delta_{xy} = 0, 2, 4,$ and 8. The number of messages was varied between 5 and 50,000. The time between generating messages at the source, and the time for servicing a message at a node, was varied between 2 and 80 milliseconds.

C. Performance of CGD

[4] Figure C shows a histogram of the number of messages handled in each node of a contour under optimal spreading in Ξ_4 and Figure C shows the uniform spreading. Notice that all the nodes in the same row of a contour handle roughly the same number of messages when CGD is used under optimal spreading. Figure 9 and figure 10 shows that the latency achieved by CGD under optimal spreading is better than the latency achieved in the other methods. Figure C and Figure C shows the throughput achieved by CGD which is better than uniform and other spreads.Strategy is to have better throughput and latency and these can be seen in the CGD with optimal spreading strategy compared to other spreads.

C.1 Effects of Link Failures

To investigate the performance of CGD under link failures, we varied the probability of link failure for the simulation scenario described above between 0 (no loss) to 1 (all messages lost). Each link in the multihop network was assumed to fail independently with equal probability.

Notice in Figure 13 and Figure 14 that although the throughput is high when for optimal spread compared to other spreads, the throughput in all the cases drops to zero when the link loss rate is about 20%. This performance is unlike the robust performance noted in the cases when each neighbor has 8 neighbors [3].

[4] Unless mentioned all the re sults are for Ξ_4

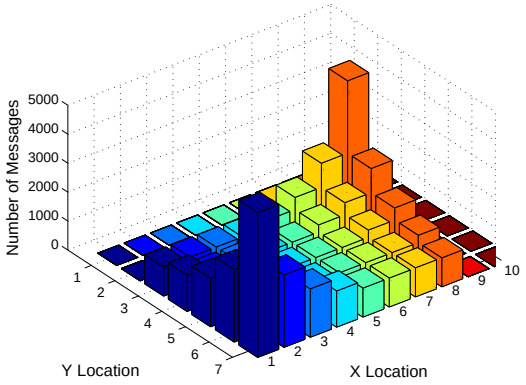


Fig. 7. All the nodes in a row handle roughly the same number of messages in CGD under optimal spreading.

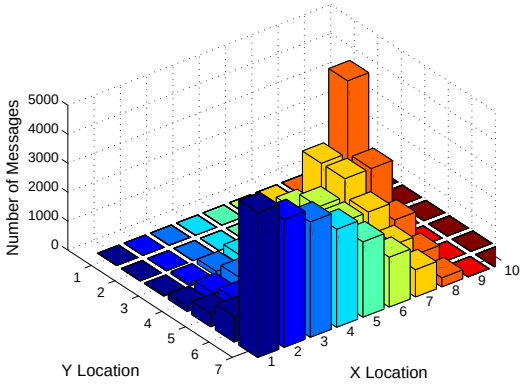


Fig. 8. Uniform Spreading will result in loading a path in set of all shortest paths.

C.2 Scalability

Because of the large difference in performance between the shortest path and CGD, we constructed an shortest paths tree to exploit the regular topology. Such a shortest path tree, i.e., an ideal shortest paths tree, may be difficult to compute for a general topology — however, we used this tree as a benchmark to compare the performance of CGD under optimal spreading. For the simulations based on the shortest paths tree, we eliminated the first hop from the simulation and all the messages from the source were initially distributed to the two immediate neighbors.

To investigate the scalability of CGD, we examined how QoS changes because of queuing effects. We increased the service time at each node and as well varied the message generation time at the source. To see the effect of messages on the contours we var-

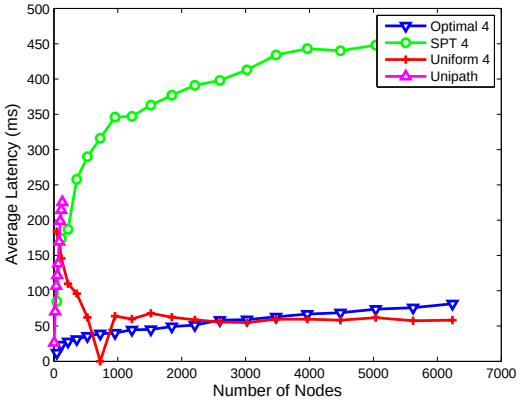


Fig. 9. Latency of CGD under optimal spreading compared to the Latency achieved using Unipath, CGD under uniform spreading, and Shortest Path Tree in Ξ_4.

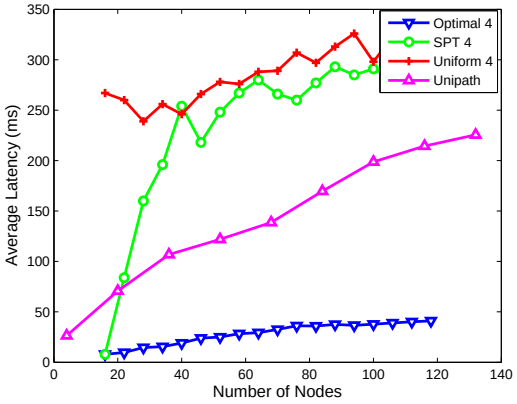


Fig. 10. Latency of CGD under optimal spreading compared to the Latency achieved using Unipath, CGD under uniform spreading, and Shortest Path Tree in Ξ_6.

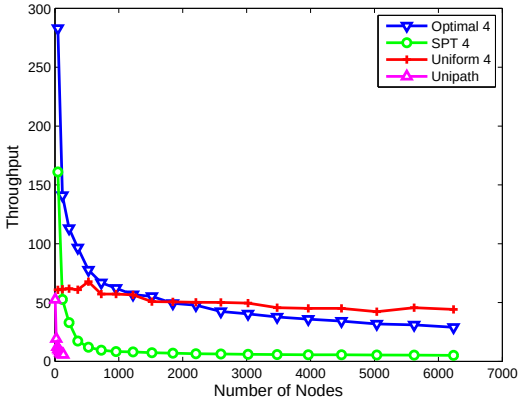


Fig. 11. Throughput achieved using CGD under optimal spreading compared to the Throughput achieved using Unipath, CGD under uniform spreading, and Shortest Path Tree in Ξ_4.

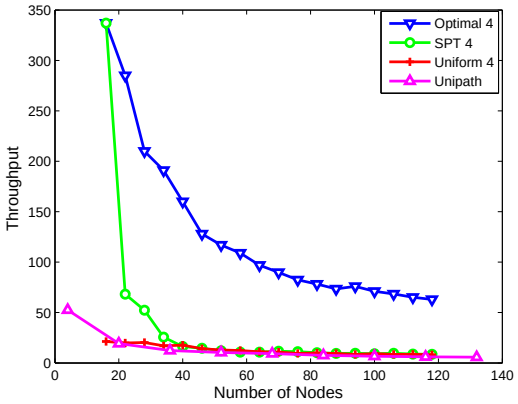


Fig. 12. Throughput achieved using CGD under optimal spreading compared to the Throughput achieved using Unipath, CGD under uniform spreading, and Shortest Path Tree in Ξ_6.

ied the number of messages in the network from 5 to 50000 which are sent from the source. Because jitter followed the trend of latency, we do not show these results here. The throughput achieved when the system scales is presented.

Figure C.2 shows the effect of slowing down the rate at which messages are serviced in each queue in Ξ_4. Because the rate at which messages are injected at the source does not change, this slow down causes the queues in the intermediate nodes to build up thereby decreasing the rate at which messages reach the sink and a fall in throughput.

Figure C.2 shows the average latency of optimal CGD when the number of messages sent by the source is increased from 5, to 50, to 500, to 5,000 and 50,000. It can be noticed, once again that average latency will be less when the number of shortest paths are more.

Figure C.2 shows the average latency of optimal CGD when the generation time at the source is varied from 2 to 80 ms and notice that the latency of the optimal spreading to be better than other spreads.

D. Summary and Discussion

The preceding results demonstrate that despite the availability of multiple shortest paths between a pair of nodes, these paths cannot be exploited using a straightforward strategy such as uniform spreading. While the results in Figure C and Figure C show that uniform spreading performs better than methods based on Unipath, these figures also show that CGD under optimal spreading effectively exploits all the available shortest paths.

Figure C shows that the nodes in the same row of a contour handle roughly the same number of messages

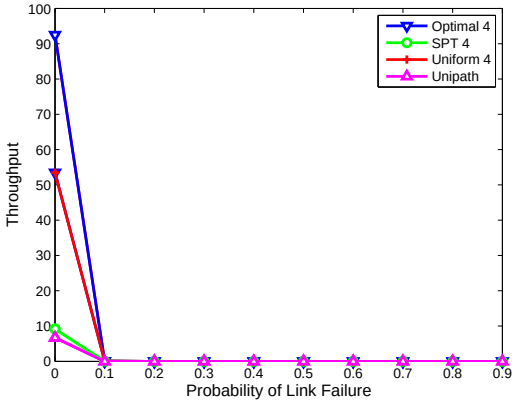


Fig. 13. Throughput of CGD under optimal spreading in in Ξ_4 .

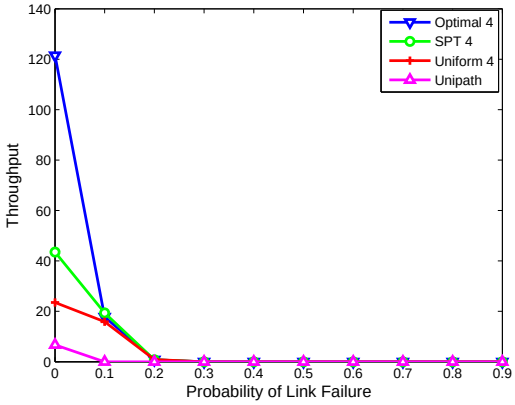


Fig. 14. Throughput of CGD under optimal spreading in in Ξ_6.

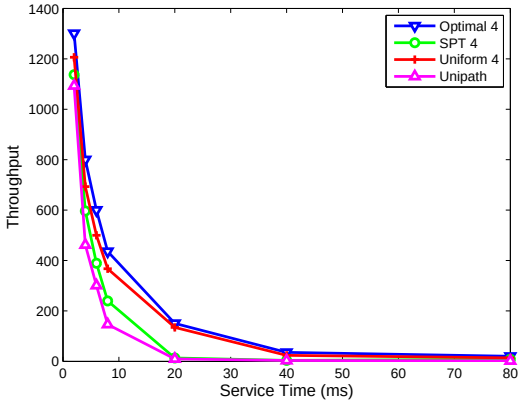


Fig. 15. Throughput variation as messages are processed more slowly - decreasing throughput.

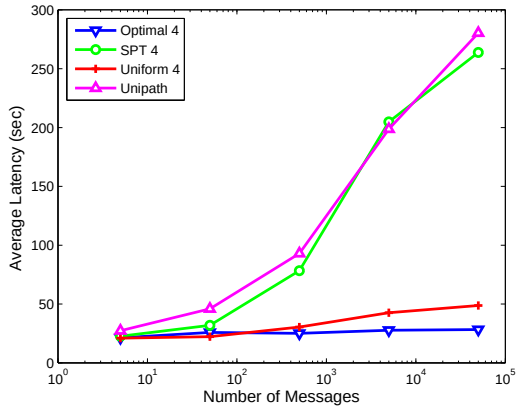


Fig. 16. Latency variation as number of messages changes.

when using CGD under optimal spreading.

The scalability results confirm that CGD under optimal spreading achieves QoS that can be achieved with a (ideal) shortest paths tree even when the number of nodes, number of messages, or the service time of each nodes increase.

VII. Conclusions

Many future engineered systems that are based on peer-to-peer connected, mesh topologies, are likely to have multiple paths between a pair of nodes. We defined a contour as the union of all shortest-paths between a pair of nodes. Using a regular topology, we proved that when the messages are spread uniformly over the paths in a contour, nodes along one path handle more messages than other messages. We presented an optimal strategy for spreading messages in such systems and results to demonstrate effectiveness of the spreading strategy. The results of this work suggest that dissemination in mesh connected topologies, with general structure, can be addressed by having some nodes spread the messages and other nodes select one of the available paths without further spreading. Identifying the extent of loading and nodes that do and do not spread the messages are interesting problems for future investigations.

References

[1] K. Akkaya and M. Younis. A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3:325–349, 2005.

[2] R.E. Bellman. *Dynamic Programming*. Princeton University press, Princeton, N.J, 1957.

[3] I-H. Chu, M. Duan, and S. Sastry. Contour guided dissemination for networked embedded systems. In *In Press. Accepted for publication in International Journal of Distributed Sensor Networks.*, Baltimore, MD, October 2007.

[4] A. Fallahi, E. Hossain, and A.S. Alfa. QoS and energy trade off in distributed energy-limited mesh/relay networks: A queuing analysis. *IEEE Transactions on Parallel and Distributed Systems*, Volume 17:576 – 592, June 2006.

[5] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient, energy-efficient multipath routing in wire-

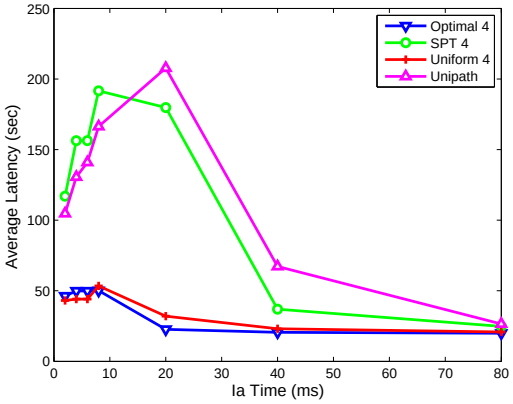


Fig. 17. Latency variation as messages are released slowly at the source.

less sensor networks. *Mobile Computing and Communications Review*, 1(2), 2002.

[6] J. Gao and L.Zhang. Load-balanced short-path routing in wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, volume 17:377 – 388, April 2006.

[7] N. Hayslip, S. Sastry, and J. Gerhardt. Networked embedded automation. *Assembly Automation*, 26(3):235–241, 2006.

[8] J. Hill, M. Horton, R. Kling, and L. Krishnamurthy. The platforms enabling wireless sensor networks. *Communications of the ACM*, 47(6):41–46, June 2004.

[9] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed diffusion for wireless sensor networking. *IEEE/ACM Transactions on Networking*, 11(1):2–16, 2003.

[10] J. Kulik, W. Heinzelman, and H. Balakrishnan. Negotiation-based protocols for disseminating information in wireless sensor networks. *Wireless Networks*, 8:169–185, 2002.

[11] S.J Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks. In *IEEE International Conference on Communications*, volume 10, 2001.

[12] K.K. Mamidisetty. Generalizing contours in mesh topologies. Master's thesis, The University of Akron, Department of Electrical and Computer Engineering, (Expected)2007.

[13] S. Muller, R.P.Tsang, and D. Ghosal. Multipath routing in mobile ad hoc networks : Issues and challenges. In *Invited paper in Lecture Notes in Computer Science*. Springer, 2004.

[14] R.A. Sahner, K.S. Trivedi, and A. Puliafito. *Performance and Reliability Analysis of Computer Systems*. Kluwer Academic Publishers, 1996.

[15] A. Vargas. Omnet++discrete event simulation system, 2003.

[16] G. Zhao, T. He, S. Krishnamurthy, and J.A. Stankovic. Impact of radio irregularity on wireless sensor networks. In *MobiSys*, pages 125–138, 2004.

Management of Sensor Networks via Graph Mining

Vijay Kumar & Praveen Rao
Dept. of Computer Science Electrical Engg.
University of Missouri-Kansas City
Kansas City, MO 64110
{kumarv,raopr}@umkc.edu

Ali Hurson
Dept. of Computer Science & Engg.
Pennsylvania State University
University Park, PA 16801
Hurson@cse.psu.edu

Sanjay Madria
Dept. of Computer Science
University of Missouri-Rolla
Rolla, MO 65409
madrias@umr.edu

Abstract

Sensor nodes have been used for some time to incorporate desired function in any functional unit. For example, sensor is used in a remote control, garage door opener, toys, etc., so that the garage door opener can open the door, the toy can move, and so on. The application of sensor nodes has recently become widespread and they are used in nearly every functional unit for performing simple to highly complex tasks. Such uses have given rise to sensor networks which could be mobile or wireless or wired and could be quite massive containing thousands of sensors. In order to perform the desired set of functions the nodes of the network must be managed efficiently. This management includes identifying dead sensors, reprogramming a set of sensors, searching faulty sensors, changing old sensors with new sensors, and so on. One of the issues here then to quickly find the desired set of sensors, which could be geographically dispersed, as quickly and accurately as possible. In this paper we introduce this issue and propose to use graph mining techniques for tracking the health and state of a sensor network.

1 Introduction

A sensor is a programmable, low-cost, low-power, multifunctional device and usually has a much shorter working life span [1]. Essentially it is highly suitable for working as a node of distributed peer-to-peer or client server architecture. A sensor node can be programmed to capture and validate the data of its environment, to perform some operation and dispatch the result to other nodes. In addition, it can also receive data from other nodes and can be queried by any peer or client of the network. For example, in a sensor network a set of sensors could be programmed to measures temperature of its surrounding. It will continue to perform this measurement unless it is reprogrammed, which can be done remotely also. Thus, the attributes and functionality of a sensor node give it a unique semantics. As a result, the nodes of a sensor network can be subjected to the set of operations applicable to a set of data objects. We can, therefore, define a set of sensor nodes as a sensor database (not database of sensors) with embedded semantics. Under this model a database technology approach can be applied to query the sensor database for finding efficiently and reliably a desired set of sensors or discovering or creating a pattern necessary for managing the massive sensor network. Thus a query such as "Find the geographical location of a set of sensors which require to be reprogrammed to monitor the emission of methane gas at a dump site" or "Find set of sensors which may fail in the next week" or "What set of sensors malfunctioned during 10:00AM to 4:00PM", etc., can be answered. In this paper we propose a scheme which uses graph mining approach for getting useful information.

2 Sensor Network Architecture

Today sensors are deployed at various places (buildings, malls, factories, high rise building, banks, etc.) of a city for continuous monitoring of events to manage security [4, 10, 12]. We define a sensor node as a triple $<$*unique function, unique property, unique ID*$>$. This triple assigns a unique semantics to a sensor node which allows us to treat

a sensor node as a data item of a database, to visualize a sensor network of any size and type (wired or wireless) as a database where data objects are sensors and change their states similar to the state change a data item goes through. This state change includes sensor node failure, sensor inconsistency (malfunctioning of a sensor), sensor manipulation (reprogramming, changing role, etc.). We identify our sensor network as "Sensorbase".

Definition 1 *Sensorbase (S) is a countably infinite set of uniquely programmed network of sensor nodes of all capability. Thus, $S = \{s_1, s_2, \cdots, s_n\}$ where s_i (i = 1, 2, $\cdots$, n) are programmed sensor nodes and s_i and s_j ($i \neq j$) may or may not be connected.*

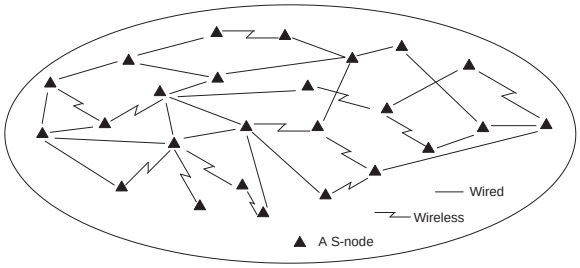


Figure 1. A massive Sensorbase.

Nodes of S captures data of its environment and dispatches it to other nodes through routers [2, 3, 6, 7, 8, 9, 11]. Figure 1 illustrates our Sensorbase which includes stationary and mobile sensor nodes.

Definition 2 *A sensor node may be mobile or stationary processing unit which is a pair <C, A>; where C is a set of its processing and communication capabilities and set of attributes $A = \{a_1, a_2, \cdot, a_n\}$; where a_i is a property set such as size, location of deployment, type, etc.*

We assume all types of sensors and each collect data about the environment it is deployed and they can go through states such as dormant (not working but alive), active, suspended, mobile, and so on. Our Sensorbase can be represented as a connected graph.

Definition 3 *A Sensorbase is a dynamically connected graph where connection between any pair of nodes can be broken and a new connection can be established anytime.*

In this platform we intend to mine knowledge about present and future states of a set of sensors. This requires searching the sensor network. One can argue that to perform this kind of search each sensor can be fitted with some notification device to announce its coordinates. This approach may work but it has a number of problems and we list a few:

1. A dead S-node cannot announce its location.
2. If every node announces its location, then this is likely to create congestion.
3. The available channels cannot support such communication traffic.
4. Identification of one or set of nodes for listening to these announcements.
5. Optimal announcement frequency.

Note that such announcements may not always be required. These problems motivated us to exploit the power of graph mining to discover the answer. We seek a need-based mining in Sensorbase to manage it efficiently and to trace an individual sensor for monitoring its functionality, repairing dead sensors, reprogramming a set of sensors, and so on.

3 Our Solution Approach

We treat a sensor node of our Sensorbase as a data item. Data in different domains can be modeled as graphs. For example, computer networks, structure of molecules, biological networks, and social networks have a graph structure. Data mining, the process of extracting useful information from large data sets, has been applied to graphs (e.g, gSpan [20], GraphMiner [21]). A key operation is to extract frequent subgraph patterns from a large database of graphs or from a large graph. Recent work by Borgwardt et.al. [22] has studied the mining of dynamic graphs, i.e., graphs change over time. Edges can be added and deleted from the graphs over time.

Sensorbase can be modeled as a dynamic, time-varying graph where the edge connectivity is transient. An edge between sensors A and B exists during a time window Δt if A and B communicate during Δt. Over time, sensors communicate with each other yielding a dynamic graph. We believe that mining such a dynamic graph will provide us useful knowledge about the health of the sensor network. In addition, we can leverage existing algorithmic solutions for mining purposes.

We believe certain communication patterns in a sensor network can provide us insight about the state and health of the sensor network. For example, a frequent communication pattern can tell us which sensors are heavily used. This may imply that these sensors may need to be recharged more often. A sensor node with high degree may indicate a potential single-point of failure. This may call for reprogramming of the sensors to reduce high node degrees. In addition, an unusual/infrequent pattern may indicate an anomalous behavior in the network. New patterns can

emerge and can help us understand the changes to the sensor network. Correlated patterns can help identify dependent sensors though they may not communicate with each other directly. This can help in understanding and localizing erroneous behaviors in the sensor network.

4 System Design and Implementation

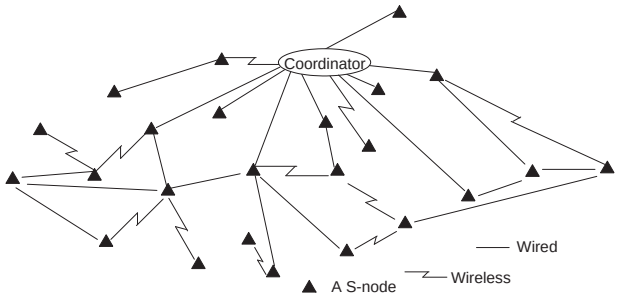


Figure 2. Structure of a *Micro-sensornet*.

A major challenge is to collect the state of sensors periodically in an efficient and scalable way. In our approach, each node of our Sensorbase generates its own application log which contains the status of the sensor. For example, in addition to operations on data (collection, aggregating, etc.) the sensor records its communication history, channel use, inactive time, sensor id, geographical location, and so on. These logs are collected, merged and preprocessed before invoking graph mining procedures for identifying desired patterns.

We structure our Sensorbase using *Micro-sensornet*, which is a small network of a set of sensors. Thus, a Sensorbase is a set of *Micro-sensornets*. A *Micro-sensornet* is created with sensor nodes which perform similar activities or which collect the same type of information (Figure 2). For example, sensors which measure temperature, moisture or pressure, etc., will be a part of a *Micro-sensornet*. Each *Micro-sensornet* has a "sink" node which serves as the *Coordinator* for all the member sensor nodes of its set. The *Coordinator* collects application logs of its members, reformats it, unifies them and includes its own log and sends it to a "server node" (Section 4.1) that performs graph mining. In addition to dispatching the log to the *Coordinator* periodically, when an event occurs, each sensor may send a control message when its status in the network changes. For example, a node may wish to leave the *Micro-sensornet* or may wish to stop collecting data, etc., then it sends a control message to its *Coordinator*. The size of a *Micro-sensornet* is decided by the number of participants and the communication in the network is established through *Micro-sensornets*. Figure 2 illustrates the structure of a *Micro-sensornet*. The participants are not visible outside of a *Micro-sensornet*.

4.1 Collecting Application Logs and Performing Graph Mining

Since sensor nodes are restricted by their computing capacity and communication power, any data-intensive mining task should be performed outside the Sensorbase. We introduce a *server node* that collects application logs from the *Coordinators* of all or a subset of *Micro-sensortnets*.

We propose two approaches for collecting logs from *Micro-sensornets* based on which either centralized graph mining or distributed graph mining will be applied. Each of these approaches use the services of *Micro-sensornet*.

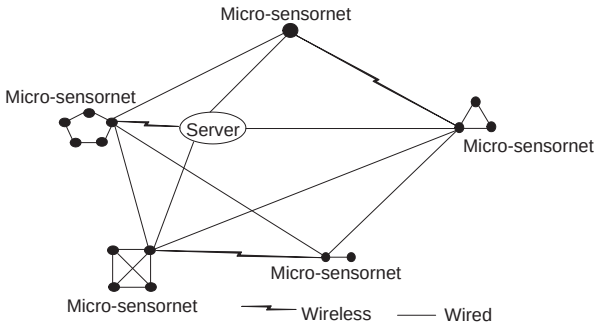


Figure 3. A centralized server approach for collecting logs and performing graph mining

Centralized Server Approach: Each *Coordinator* dispatches its collected log to the same server node as shown in Figure 3. The server merges the logs from different *Coordinators* and constructs a graph for the time window specified in the logs. The server may need to detect and remove redundant information from these logs. The server executes graph mining algorithm to discover interesting patterns/trends. The results can be presented to a human administrator to take corrective actions. A centralized approach is simpler by design. However, it can suffer from scalability problems.

Distributed Server Approach: To make the log collection and mining process scalable, we propose a distributed server architecture, which consists of a set of server nodes that collect logs and perform mining. A *Micro-sensornet* sends its log to a server based on membership, availability, bandwidth, and distance (for wireless communication). The group of servers together run a distributed mining algorithm to discover interesting patterns/trends. The servers organize themselves in an overlay network and collectively maintain the dynamic graph for the entire Sensorbase.

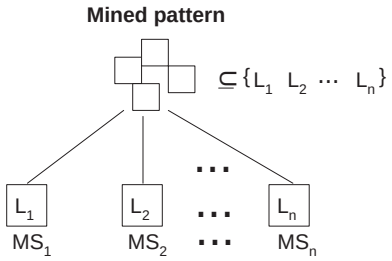


Figure 4. Graph mining result

It would be necessary to reformat the raw data (application log) for the mining routine. In distributed approach there is no central repository, so a "virtual" log unification would be necessary. We propose to use the services of *mobile* agents for this unification. Under this approach each *Micro-sensornet* will have a mobile agent which receives application log from each participants, filters it, and performs aggregation (union with other reformatted log). The mining routine also has a set of mobile agents which are shipped to coordinators where they examine the formatted data and perform partial mining. When the desired pattern is discovered from a subset of coordinators, they are combined together to get the final result. Figure 4 sketches our mining approach. The final mined pattern is a subset of *Micro-sensornets* logs. For example, to mine the pattern for *Find the geographical location of a set of sensors which require to be reprogrammed to monitor the emission of methane gas at a dump site* the mining routine agents will discover, communicate, and visit only those coordinators which are responsible for collecting this data.

5 Conclusions and Future Work

We have presented the basic framework of a system for managing large sensor networks by applying graph mining techniques. A Sensorbase consists of a number of *Micro-sensornets* each consisting of several sensors performing similar tasks. A Sensorbase can be modeled as a dynamic, time-varying graph where the edge connectivity (based on communication between two sensors) is transient. By collecting and analyzing logs from coordinators, the state of the graph is maintained by one server or a group of servers. By mining the communication patterns of the Sensorbase, we believe that insightful knowledge about its health and state can be obtained. In this paper, we only present the basic design of our system. We understand that there are many challenges to address. As part of future work, we will develop the mining routine and build a prototype system to demonstrate the utility and feasibility of our proposed approach.

References

[1] Estrin, R. Govindan and J. Heidemann, "Embedding the Internet", *ACM Comm.*, Vol. 43, No. 4, May 2000.

[2] Alberto Cerpa and Deborah Estrin, "Ascent: Adaptive Self-Configuring sensor Network Topologies," Technical Report UCLA/CSD-TR-01-0009 Computer Science Department, UCLA.

[3] T. Clouqueur, P. Ramanathan, K. K. Saluja, and K-C. Wang, "Value-fusion versus decision-fusion for fault-tolerance in collaborative target detection in sensor networks," *Proceedings of the Fourth International Conference on Information Fusion*, August 2001.

[4] H. Deng, W. Li, and Dharma P. Agrawal, "Routing Security in Ad Hoc Networks," IEEE Communications Magazine, Special Topics on Security in *Telecommunication Networks*, Vol. 40, No. 10, October 2002.

[5] EPA publication, "Nonhazardous waste landfill publications," CR-ROM, EPA530-C-00-002.

[6] Soheil Ghiasi, Ankur Srivastava, Xiaojian Yang, and Majid Sarrafzadeh, "Optimal Energy Aware Clustering in Sensor Network," *Sensors*, Vol. 2, July 2002.

[7] W. B. Heinzelman and A.P. Chandrakasan and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Transactions on Wireless Communications*, Vol. 1, No. 4, October 2002.

[8] Chalermek Intanagonwiwat and Ramesh Govindan and Deborah Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking* (MOBICOM), August 2000

[9] Stephanie Lindsey, Cauligi Raghvendra, and Krishna Sivalingam, "Data Gathering Algorithms in Sensor Networks Using Energy Metrics," *IEEE Trans. on Parrallel and Distributed Sys*. 13, No. 9, September 2002.

[10] Perrig, P. Szewczyk et at., "SPINS: Security Protocols for Sensor Networks," *Wireless Networks* 8, 2002.

[11] R. Viswanathan and P. K. Varshney, "Distributed detection with multiple sensors: Part I - fundamentals," *Proceedings of the IEEE*, Vol. 85, No. 1, January 1997.

[12] D. Wood and J. A. Stankovic, "Denial of Service in Sensor Networks," *IEEE Computer*, 35(10), Oct. 02.

[13] Deborah Estrin, Ramesh Govindan, John Heidemann, Satish Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks," In *Proceedings of the Fifth Annual International Conference on Mobile Computing and Networks* (MobiCOM '99), August 1999, Seattle, Washington.

[14] J. Girao, D. Westhoff, M. Schneider, "CDA: Concealed Data Aggregation for Reverse Multicasst Traffic in Wireless Sensor Networks," In *Proccedings of IEEE ICC*, 2005.

[15] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of network density on data aggregation in wireless sensor networks," In *Proceedings of International Conference on Distributed Computing Systems*, November 2001.

[16] W. Jonker and M. Petkovic, Using secret sharing for searching in encrypted data, *Workshop on Secure Data Management in a Connected World* (SDM), LNCS 3178, Toronto, Canada, Aug. , 2004

[17] Chris Karlof, Naveen Sastry, and David Wagner, "Tinysec: A link layer security architecture for wireless sensor networks," In *Second ACM Conference on Embedded Networked Sensor Systems* (SensSys 2004), November 2004.

[18] B. Przydatek, D. Song, and A. Perrig, "SIA: Secure Information Aggregation in Sensor Networks," In *Proc. of ACM SenSys* 2003.

[19] Radu Sion, M. J. Atallah, Sunil Prabhakar, "Resilient Rights Protection for Sensor Streams," *Proc. of the International Conference on Very Large Databases* (VLDB), pp. 732-743, 2004.

[20] Xifeng Yan, Jiawei Han, "gSpan: Graph-Based Substructure Pattern Mining," *Proceedings of the 2002 IEEE International Conference on Data Mining* (ICDM '02),pages 721-724, 2002

[21] Wei Wang, Chen Wang, Yongtai Zhu, Baile Shi, Jian Pei, Xifeng Yan, Jiawei Han, "GraphMiner: a structural pattern-mining system for large disk-based graph databases and its applications," *SIGMOD* Conference 2005: 879-881

[22] Karsten M. Borgwardt, Hans-Peter Kriegel, Peter Wackersreuther, "Pattern Mining in Frequent Dynamic Subgraphs," *Proceeding of the 6th IEEE International Conference on Data Mining* (ICDM'06), pages 818-822, 2006

Role of IBM's SSME discipline in Cyberspace operations

Ramesh Reddi
Managing Consultant
IBM Global Business Services

***Abstract*—The key emphasis of the United States Cyberspace operations is to protect the Critical Infrastructure Protection (CIP). CIP is a national program to assure the security of vulnerable and interconnected infrastructures of the United States. This recognizes certain parts of the national infrastructure as critical to the national and economic security of the United States and the well-being of its citizenry. The national infrastructure components are physical and virtual systems that are so vital to the United States that the incapacity or destruction of such systems and assets would have debilitating impact on security, national economic security, national public health or safety. The CIP classifies the infrastructure into sectors. The defined sectors are Banking and finance, Transportation, Power, Information and Communications, Federal and municipal services, Emergency services, Fire departments, Law enforcement agencies, Public works, Agriculture and food, and National monuments and icons.**

Critical Infrastructure Protection is therefore a nationwide program, with government operating in partnership with private industry. The program includes a national structure and a National Infrastructure Assurance Plan.

IBM recently developed a new academic discipline called SSME (Services sciences, Management and Engineering). SSME hopes to bring together ongoing work in computer science, operations research, industrial engineering, business strategy, management sciences, social and cognitive sciences, and legal sciences to develop the skills required in a services-led economy.
This presentation discusses how IBM's SSME can be useful in cyberspace operations such as protecting Critical Infrastructure of the country. Our National Strategy for the CIP involved two components. The first component of the National Strategy is to develop initiatives to satisfy the cross-sector security priorities. The second component of the National Strategy involves addressing issues that relate to unique protection areas. The paper sheds light on IBM's vision of SSME and its impact in developing these initiatives especially cross-sector security priorities to protect the critical infrastructure sectors and the vital services they provide to the country.

***Index Terms*—Critical Infrastructure Protection (CIP), Systems Science, Management and Engineering (SSME), Information Sharing Access Centers (ISAC), Mathematical modeling, Service systems, Sectors, National Strategy, Cross-sector Security priorities**

Ramesh Reddi is managing consultant with IBM Global Business Services e-mail: rreddi@us.ibm.com

.

I. Critical Infrastructure Protection Overview

Numerous officials within the public and private sectors of the United States have been actively promoting and applying critical infrastructure protection (CIP). The urgent call for the protection of critical infrastructures began on 11 September 2001, when leaders of government and industry as well as millions of private citizens were awakened from their slumber of national safety and security.

Homeland Security Presidential Directive – 7 (HSPD-7) issued in December 2003 established the policy of the United States to enhance the protection of national critical infrastructures against terrorist acts that would significantly diminish the responsibility of federal, state, and local governments to perform essential security missions and to ensure the general public health and safety. The USA PARTRIOT Act of 2001 defines critical infrastructures as "those physical and cyber-based systems so vital to the operations of the United States that their incapacity or destruction would have a debilitating impact on national defense, economic security, or public safety." More specifically, critical infrastructures are those people, things, or systems that must be intact and operational in order to make daily living and working possible.

The term "critical infrastructure protection" (CIP) pertains to the proactive activities for protecting critical infrastructures: the people, physical assets, and communication/cyber systems that are indispensably necessary for national security, economic stability, and public safety. CIP methods and resources deter or mitigate attacks against critical infrastructures caused by people, by nature, and by HazMat accidents. Plainly stated, CIP is about protecting those invaluable assets that make life, liberty, and the pursuit of happiness a national reality.

Community leaders, including those of emergency first responders, have the responsibility to decide which infrastructures must be protected from all hazards. Scarce resources (i.e., time, money, personnel, and material) make these decisions somewhat complicated. The government recommends the implementation of the CIP process.

The CIP process is an analytical model or template to guide the systematic protection of critical infrastructures. More basically, it is a reliable decision sequence that assists leaders

in ultimately determining exactly what really needs protection as well as when the protection should be activated. As a time-efficient and resource-restrained practice, the process ensures the protection of only those infrastructures upon which survivability, continuity of operations, and mission success depend. The process consists of the following steps:

- Identifying critical infrastructures essential for mission accomplishment.
- Determining the threats against those infrastructures.
- Analyzing the vulnerabilities of threatened infrastructures.
- Assessing the risks of the degradation or loss of a critical infrastructure.
- Applying countermeasures where risk is unacceptable.

When applied by the emergency services, CIP is not a product; it is a process to secure the effective protection of mission critical people and systems. While it may be impossible to prevent all attacks against critical infrastructures, CIP can reduce the chances of future attacks, make it more difficult for attacks to succeed, and mitigate the outcomes in the event they do occur. Thus, among all the important procedures or things involved in emergency preparedness, CIP is possibly the most essential component.

II. Sector Description

The following are the sectors that comprise of the above national infrastructure:

Agriculture and food

Provides for the fundamental need for food. The infrastructure includes supply chains for feed and crop production. Carries out the post harvesting of the food supply, including processing and retail sales. Managed by Departments of Agriculture, and Health and Human Services, Food and Drug Administration.

Defense and Industrial Base

Supplies the military with the means to protect the nation by producing weapons, aircraft, and ships and providing essential services, including information technology and supply and maintenance. Managed by Department of Defense.

Energy

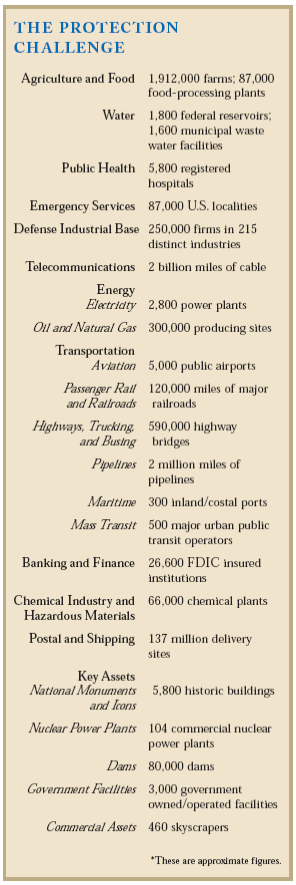
Provides the electric power used by all sectors and the refining, storage, and distribution of oil and gas. The sector is divided into electricity and oil and natural gas. Managed by Department of Energy.

Public Health and health care

Mitigates the risk of disasters and attacks and also provides recovery assistance if an attack occurs. The sector consists of health departments, clinics, and hospitals. Managed by Department of Health and Human Services.

National Monuments and icons

Memorializes or represents monuments, physical structures, objects, or geographical sites that are widely recognized to represent the nation's heritage, traditions, or values, or widely recognized to represent important national cultural, religious, historical, or political significance. Managed by Department of the Interior.



THE PROTECTION CHALLENGE

Agriculture and Food	1,912,000 farms; 87,000 food-processing plants
Water	1,800 federal reservoirs; 1,600 municipal waste water facilities
Public Health	5,800 registered hospitals
Emergency Services	87,000 U.S. localities
Defense Industrial Base	250,000 firms in 215 distinct industries
Telecommunications	2 billion miles of cable
Energy	
Electricity	2,800 power plants
Oil and Natural Gas	300,000 producing sites
Transportation	
Aviation	5,000 public airports
Passenger Rail and Railroads	120,000 miles of major railroads
Highways, Trucking, and Busing	590,000 highway bridges
Pipelines	2 million miles of pipelines
Maritime	300 inland/costal ports
Mass Transit	500 major urban public transit operators
Banking and Finance	26,600 FDIC insured institutions
Chemical Industry and Hazardous Materials	66,000 chemical plants
Postal and Shipping	137 million delivery sites
Key Assets	
National Monuments and Icons	5,800 historic buildings
Nuclear Power Plants	104 commercial nuclear power plants
Dams	80,000 dams
Government Facilities	3,000 government owned/operated facilities
Commercial Assets	460 skyscrapers

*These are approximate figures.

Figure 1 Comprehensive view of National Infrastructure

Drinking water and water treatment systems

Provides sources of safe drinking water from more than 53,000 community water systems and properly treated wastewater from more than 16,000 publicly owned treatment works. Managed by Environmental Protection Agency.

Chemical

Transforms natural raw materials into commonly used products benefiting society's health, safety, and productivity. The chemical sector produces more than 70,000 products that are essential to automobiles, pharmaceuticals, food supply, electronics, water treatment, health, construction, and other necessities. Managed by Office of Infrastructure Protection.

Commercial Facilities

Includes prominent commercial centers, office buildings, sports stadiums, theme parks, and other sites where large numbers of people congregate to pursue business activities, conduct personal commercial transactions, or enjoy recreational pastimes. Managed by Office of Infrastructure Protection.

Dams

Manages water retention structures, including levees, more than 77,000 conventional dams, navigation locks, canals (excluding channels), and similar structures, including larger and nationally symbolic dams that are major components of other critical infrastructures that provide electricity and water. Managed by Office of Infrastructure Protection.

Emergency Services

Saves lives and property from accidents and disaster. This sector includes fire, rescue, emergency medical services, and law enforcement organizations. Managed by Office of Infrastructure Protection.

Nuclear reactors, materials and waste

Provides nuclear power, which accounts for approximately 20 percent of the nation's electrical generating capacity. The sector includes commercial nuclear reactors and non-power nuclear reactors used for research, testing, and training; nuclear materials used in medical, industrial, and academic settings; nuclear fuel fabrication facilities; the decommissioning of reactors; and the transportation, storage, and disposal of nuclear materials and waste. Managed by Office of Infrastructure Protection.

Information Technologies

Produces information technology and includes hardware manufacturers, software developers, and service providers, as well as the Internet as a key resource. Managed by Office of Cyber Security and Communications.

Communications

Provides wired, wireless, and satellite communications to meet the needs of businesses and governments. Managed by Office of Cyber Security and Communications.

Postal and shipping

Delivers private and commercial letters, packages, and bulk assets. The U.S. Postal Service and other carriers provide the services of this sector. Managed by Transportation Security Administration.

Transportation systems

Enables movement of people and assets that are vital to our economy, mobility, and security with the use of aviation, ships, rail, pipelines, highways, trucks, buses, and mass transit. Managed by Transportation Security Administration and U.S. Coast Guard.

Government Facilities

Ensures continuity of functions for facilities owned and leased by the government, including all federal, state, territorial, local, and tribal government facilities located in the United States and abroad. Managed by Immigration and Customs Enforcement, Federal Protective Service.

II NATIONAL STRATEGY FOR CIP

A. Cross-sector Security priorities

This strategy identifies major cross sector initiatives in the following areas:

1. Planning and Resource Allocation
   - Create collaborative mechanisms for government-industry critical infrastructure and key asset protection planning
   - Identify key protection priorities and develop appropriate supporting mechanisms for these priorities;
   - Foster increased sharing of risk-management expertise between the public and private sectors;
   - Identify options for incentives for private organizations that proactively implement enhanced security measures;
   - Coordinate and consolidate federal and state protection plans;
   - Establish a task force to review legal impediments to reconstitution and recovery in the aftermath of an attack against a critical infrastructure or key asset;
   - Develop an integrated critical infrastructure and key asset geospatial database; and
   - Conduct critical infrastructure protection planning with our international partners.
2. Information Sharing and Indications and Warnings
   - Define protection-related information sharing requirements and establish effective, efficient information sharing processes;
   - Implement the statutory authorities and powers of the Homeland Security Act of 2002 to protects security and proprietary information regarded as sensitive by the private sector;
   - Promote the development and operation of critical sector Information Sharing Analysis Centers;
   - Improve processes for domestic threat data collection, analysis, and dissemination to state and local government and private industry; Support the development of interoperable secure communications systems for state and local

governments and designated private sector entities; and Complete implementation of the Homeland Security Advisory System.

3. Personal Surety, Building Human Capital and Awareness
   - Coordinate the development of national standards for personnel surety;
   - Develop a certification program for background screening companies;
   - Explore establishment of a certification regime or model security training program for private security officers;
   - Identify requirements and develop programs to protect critical personnel;
   - Facilitate the sharing of public- and private-sector protection expertise; and
   - Develop and implement a national awareness program for critical infrastructure and key asset protection.
4. Technology, and Research and Development
   - Coordinate public- and private-sector security research and development activities;
   - Coordinate interoperability standards to ensure compatibility of communications systems;
   - Explore methods to authenticate and verify personnel identity; and
   - Improve technical surveillance, monitoring and detection capabilities.
5. Modelling, Simulation and Analysis
   - Enable the integration of modeling, simulation, and analysis into national infrastructure and asset protection planning and decision support activities;
   - Develop economic models of near- and long-term effects of terrorist attacks;
   - Develop critical node/chokepoint and interdependency analysis capabilities;
   - Model interdependencies across sectors with respect to conflicts between sector alert and warning procedures and actions;
   - Conduct integrated risk modeling of cyber and physical threats, vulnerabilities, and consequences;
   - Develop models to improve information integration.

B. Unique Protection areas

The strategy gives guidelines to develop actions to address the unique issues in the protection of critical infrastructure related to individual sectors. They are categorized as below:

1. Securing Critical Infrastructure Sectors
2. Protecting Key assets

II IBM's SSME

The concept of Services Sciences, Management and Engineering (SSME) stepped into the limelight when IBM's CEO and chairman, Samuel Palmisano, published an article in the U.S. Council on Competitiveness journal Innovate America, in which he called for promoting research into "service science." SSME embodies this, as it is the fusion of the pre-existing fields of computer science, operations research, industrial engineering, mathematics, management sciences, decision-making theory, social and cognitive sciences, and legal sciences.

SSME also contributes to systematic innovation and improved productivity, and is the guiding force for the improvement of services through improved predictability in the productivity, quality, performance, compliance, development, reusability of knowledge, and operational innovation in services. Further, SSME probes the value of service providers and clients within collaborative activities and risk sharing

Specifically, SSME refers to the following:

- Mathematical modelling of service systems, and the social sciences that are relevant to understanding the human, organizational, and cultural aspects of service systems.
- Understanding of the origins and life cycles of service systems, ranging from business components, to business models, to value networks of many businesses linked globally.
- The design, development, deployment, operations, and maintenance of service systems based on IT, knowledge workers, outsourced organizational or business components – all configured to co-create, deliver, and capture value between a provider and a client.

The SSME discipline can be viewed as below:

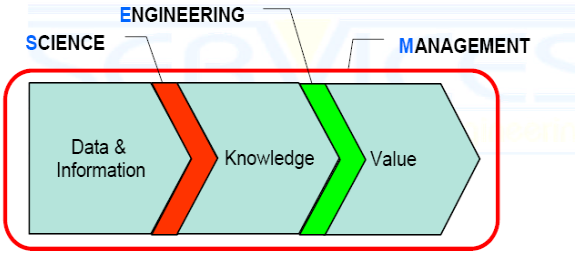


Figure 2 SSME's sub-disciplines

In the following the inter-disciplinary area of SSME is shown. It involves technology innovation, business innovation, social innovation and demand innovation.

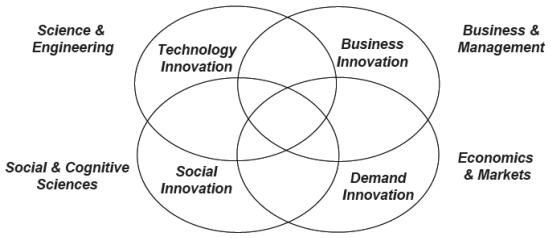


Figure 3 Inter-disciplinary nature of SSME

III SSME'S ROLE IN CIP AND EXAMPLES

Majority of the initiatives listed in the National Strategy results in some kind of process or supply chain. Or in other words a Service System. These service systems can be understood thru developing the flow diagrams, mathematical modeling, and performing the life cycle operations of these systems. The key emphasis is that these service systems model both the human and technological interaction.

The following gives a pictorial picture of what SSME is:

Figure 4 High Level view of SSME's work flow

Few examples are given to expand these ideas.

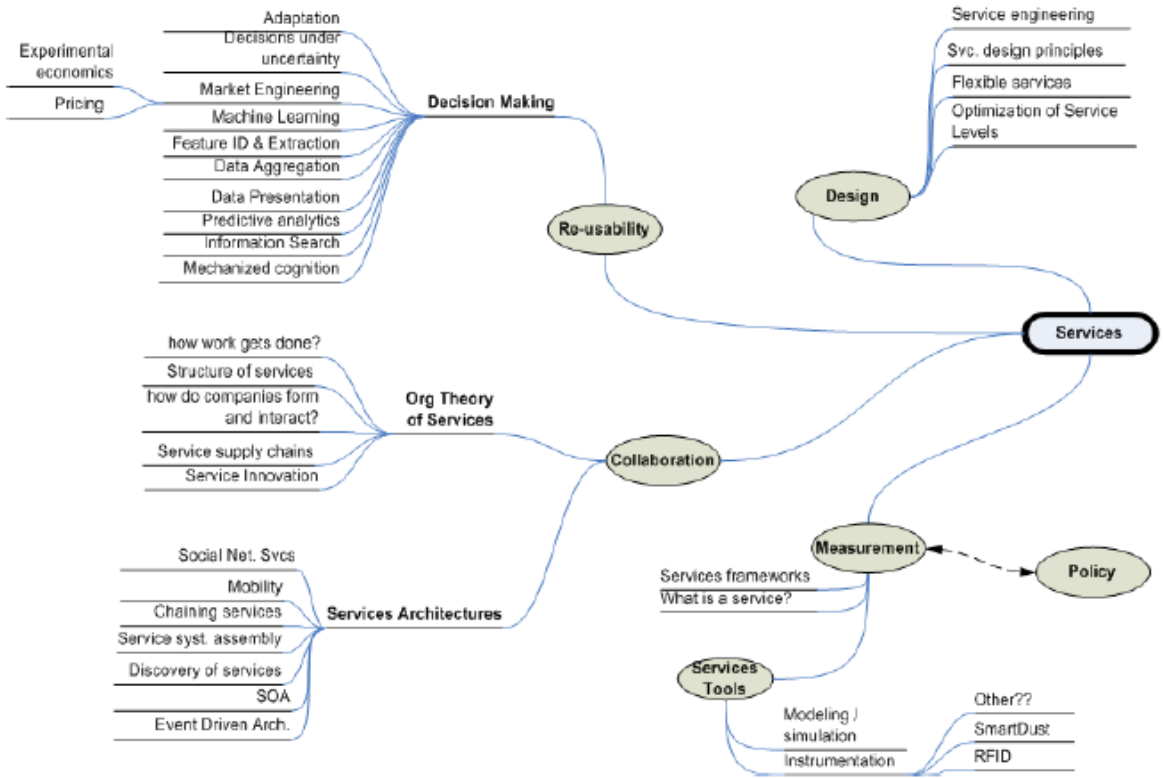
A. Risk Modeling

Risk modeling is mentioned as one of the cross-sector priority above. There is a need to develop risk models and conduct integrated risk assessments of cyber, physical threats and vulnerabilities.

These assessments include threat analysis to provide a baseline and frame of reference for risk management and investment decisions. This analysis, coupled with vulnerability assessments determine the effectiveness of security systems and tools and will provide information on critical assets and nodes. Such studies would comprise models of security incidents involving various types of both cyber and physical attacks. Analysis will focus on the complex interactions between physical and cyber systems to determine the full range of potential consequences and to ensure the applicability of findings across infrastructures.

B. Information Sharing Analysis Centers (ISAC)

The government has built sector based Information Sharing Analysis Centers (ISACs) to share the information between public and private sectors. These structures are used to communicate potential risks, threats, vulnerabilities, and incident data. The operation of these ISACs and managing the interactions of these ISACs involves completely using SSME principles.



C. Planning and Resource Allocation

The critical infrastructures need to be planned and coordinated. During that process, the resources need to be allocated to various organizations in Federal, State and Local governments. Some of the tasks are: Define clearly their critical infrastructure and key asset protection objectives; Develop a business case for action to justify increased security investments; Establish security baselines, standards, and guidelines; and Identify potential incentives for security-related activities where they do not naturally exist in the marketplace. All these areas fall under the SSME discipline.

D. Information Integration

The integration of threat and vulnerability information between sectors needs to be modeled, as well as information sharing between the federal government and critical infrastructures, to identify points of inefficiency and information loss.

IV CONCLUSIONS

The new academic discipline, SSME (Service Science, Management and Engineering) is described. The elements that comprise this new field have direct application in the area of the Critical Infrastructure Protection.

V REFERENCES

1. "The National Strategy for the physical protection of critical infrastructures and key assets" US Government publication Feb 2003
2. IBM's SSME web site (http://www.research.ibm.com/ssme)
3. "Service Science: The next frontier in service innovation" An IBM Publication
4. SSME Conference 2006: Organized by IBM Almaden Research Center
5. SSME Presentation by Shankar Sastry, Professor @ University of California, Berkeley
6. SSME @ North Carolina State University (http://www.ssme.ncsu.edu)
7. "IBM Takes the Guesswork Out of Services Consulting" By Ron Hira and Harry Goldstein, December 2006, IEEE Spectrum

Ramesh Reddi is a managing consultant with IBM Global Business Services. He is a US citizen and lives in Raleigh, NC, USA. He is a certified PMP from Project Management Institute and an experienced software business process architect with over 15 years of experience in software project management, IT application delivery, customer advocacy, business processes, On Demand, high availability, and security architecture. He also holds CISA, CISSP, and ITIL/COBIT Foundations certificates. Ramesh had collaborated with universities in the past to bid and execute Federal grants and projects. He was awarded DARPA contracts and executed them in collaboration with UCONN.

Ramesh holds multiple graduate degrees including an MBA in High Technology Management. In addition to his IBM business experience, he acquired entrepreneurial experience through founding and operating an IT services firm in USA

Stream Hierarchy Data Mining for Sensor Data

Margaret H. Dunham
Dept of Computer Sc. and Engineering
Southern Methodist University
Dallas, Texas 75275
mhd@engr.smu.edu

Vijay Kumar
Dept of Computer Sc. And Engineering
University of Missouri-Kansas City
Kansas City, Missouri 64110
kumarv@umkc.edu

Abstract. *We propose a hierarchical approach to managing stream data created by sensors. This new technique facilitates implementation of diverse software solutions to the many different types of data and requirements presented by sensor systems. At the same time it facilitates software reuse for many of the individual components of the system.*

1. INTRODUCTION

A sensor is a programmable, low-cost, low-power, multi-functional device which captures and sends data about the environment in which it is deployed [EGH00]. For example, they can be embedded strategically at various geographical points in a flood-prone area for collecting necessary data to effectively deal with flooding, to minimize its effect, and forecast its future behavior [EGHK99]. They collect data from earth-orbiting satellites such as NASA's EOSDIS (Earth Observing System Data and Information System) project. Sensors are also deployed at places unreachable by humans such as deep sea bed, enemy territory, etc. Each environment has its own type of data which requires a data-specific processing scheme. It is not possible to device a common technique to deal with all types of sensor data. A decade in the past the database community tried unsuccessfully to apply database approaches to deal with satellite data [SFGJ93, D92, CFGR92]. Later, as database technology evolved, satellite data was successfully managed by several off-the-self software tools developed for object-relational database systems. This brief history of dealing with sensor data indicates that it has some unique properties and requires special processing approaches. In this paper we investigate this issue and propose a new design hierarchy. This *Stream Hierarchy* facilitates implementation of diverse software solutions to the many different types of data and requirements presented by sensor systems. At the same time it facilitates software reuse for many of the individual components of the system.

Data captured and sent by a set of sensors is usually referred to as "stream data". It is a real-time sequence of encoded signals which contain desired information. It is continuous, ordered (implicitly by arrival time or explicitly by timestamp or by geographic coordinates) sequence of items [GO03] which is often obtained through elaborate sensor systems perhaps of many different types. Stream data which is related to event such as airplane crash, spaceship landing, etc., is strictly temporal and its validity or usefulness may be short-lived. In addition every stream data is infinite: the data keeps coming. For these reasons the management of this spatio-temporal data is very difficult and complex.

A number of processes such as cleaning, aggregating, storing, etc., are required before stream data can be used at the user level. Consider the case of stream data about flooding. Questions such as "*how frequently should we store data: every second, every minute, or every hour*?" must be resolved. Similarly, consider a video sensor with multiple resolution capability. Questions such as "*what is the resolution at which the stream data of video should be sent, how many pictures should be stored per second*?" [M03]. If we consider stream data for temperature, then we need to decide similar parameters for capturing and storing data. If the temperature variation is rare, then it does not make sense to send data continuously. On the other hand if the variation is fast, then interval should be quite small otherwise one has to find a way of filling up missing temperature reading. Note also that the amount and type of data to capture and store depends upon the application which will use the data. For real-time querying, it may not be necessary to store the entire stream data since the query may be very specific and could require only a tiny subset. On the other hand, for other purposes such as for statistical analysis, prediction, visualization, etc., the entire stream (or a model thereof) must be stored.

These issues arise in every type of stream data and their management is application-dependent. The important point is that at the user level the semantics of stream data must be clear because the interpretation of data is often performed by domain experts who need to quickly make intelligent decisions concerning the low level sensor data. For example, an employee at NOAA may be observing sensor data output concerning river flow or precipitation to make predictions of flood warning or flood watch A new system used by NOAA, *Advanced Hydrologic Prediction Service (AHPS*), targets the quick dissemination of potential flooding problems (http://www.nws.noaa.gov/oh/ahps/index.html). Sensors used to provide input to this

system include precipitation, river level and flow rate gauges, radar, snow cover and melt data. To assist users in making "risk based decisions" many visualization tools are provided [NWS07]. A NOAA employee working to make flood predictions is not a sensor or data mining expert. Thus he needs to be provided with information at the level with which he can make intelligent decisions. This actionable intelligence is thus at a very high level and quite removed from the low level sensors where the raw data is captured.

One of the challenges is capturing data samples from the hardware. In [HHM03] authors presented (a) declarative acquisition, (b) asynchronous acquisition, and (c) life-time-based acquisition. In the declarative approach a query language defines the acquisition approach [MFHH03]. It decides, unlike database systems, when and where the required data should be captured. The capturing process can be reconfigured dynamically to minimize the presence of redundant data. In approach (b) data is captured when some condition is satisfied and in approach (c) the user defines a data capture time duration. This scheme allows an adjustable data collection process where depending upon some criteria the collection rate is varied. This phase is crucial because all higher levels of stream data processing depend upon the raw data.

To monitor any event a large number of sensors is required and they must work together. For example, a large number of sensors collect flooding data for a particular target location. The same reading may be performed by these sensors and after applying some processing the raw data may be sent to a "sink" node. To increase data accuracy and improve its information content, the sink node may filter, reformat, and apply some statistical operations on the data. This operation is usually called data aggregation, which is an essential step for minimizing communication and data processing cost. In reality, the stream data passes through a number of operation-layers before it is given to users for high level analysis.

In the last decade, a number of areas in sensor research such as data aggregation, secured data model, data stream management, etc., have been actively

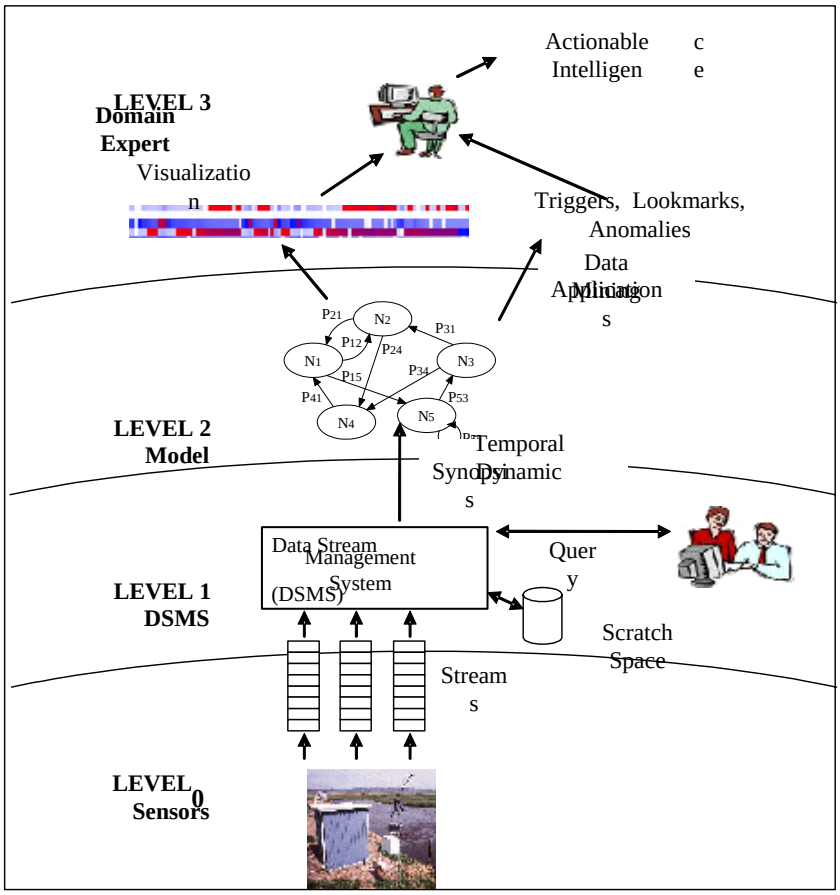


Fig. 1. Data Stream Hierarchy

investigated. Among these it appears that most progress has been made in the development of *Data Stream Management Systems (DSMS)* [AAB05, ACC03, BBDMW02, HMA04, ABBCDIMSW04, DGRTY03] for querying and managing such data. Stream data query mechanisms allow one to query fast stream data to get the most recent information from the stream and offers timely information for modern database applications. Data aggregation seems to be the second topic of choice. It defines a paradigm for routing in sensor networks [HEI01, IGE00] and helps to minimize or eliminate data redundancy and saves power. Some work has also been in the area of visualizing stream data [ZAI, GG06], which is a relatively new area of research. Visualization displays relationship among data streams which is not evident in raw data. It is needed to observe in real-time the spatial and temporal patterns and trends hidden in the data. This is a highly useful approach for fast data analysis and prediction.

Each of these earlier works has concentrated mainly on one aspect of the overall problem: some researchers have looked only at data capturing [HHM03, ECHBB, MID07], others have looked at data streams [GO03], while others have concentrated on visualization or summarization tools of the low level sensor data [ZAI, GG06]. This is similar to the state of database processing in the early 1970s. Our work is stimulated by these advances and in this paper we provide an overview of a new hierarchical view of data stream mining. What is needed is a higher level approach to processing sensor data. This approach provides levels of abstraction in processing the sensor data and actually incorporates functionality at all levels of the sensor data management – from the low level generation of the raw data to the high level presentation of the data to the domain expert who needs to make decisions and recommendations concerning the data. There is simply too much data to assume that individuals at this level will be continually making queries to the sensor/stream data. Instead, in our model we envision that push based applications are used to send data to the domain expert. This pushed data provides the information needed to make decisions.

2. PROPOSED STREAM HIERARCHY

We indicated earlier that a number of refinements and revisions are applied to the raw data before it can be useful to top level users. We propose a four level data abstraction to facilitate the creation of actionable intelligence for domain experts evaluating sensor data. This coordinated hierarchical view facilitates access to the data at each level. When viewed level by level, you can see the conventional views of the data as has been studied in the past. What is new, then, is not the level by level view of the data or functionality. Rather the overall view as a consolidated model. This can be viewed somewhat like the traditional database data abstraction levels. The traditional view of data abstraction consists of three levels: physical, logical, and external view. At the lowest level, physical, the data is actually stored. The logical level provides a higher level view of the data which is independent of how it is stored. In addition it provides data semantics and data relationships. Different users of the data may have different views at the external level. *Database Managements Systems (DSMS)* facilitate this hierarchical view of the data by the software provided. Data streams can not be easily viewed in this manner partly due to the volume of data and partly to the continuous arrival of data. There is too much to store. We also feel that the temporal aspect of the data is itself part of the data. Most previous data stream synopsis techniques have either ignored the temporal relationships or have down played them.

As seen in Figure 1, our hierarchical stream data model has four levels:

- **Level 0 - Physical Level**: This is level where the raw data is generated. We assume that sensors are used to obtain the data. The sensors are placed at many sites and they may move. The raw data at this level may or may not be actually stored.
- **Level 1 – DSMS**: At this level the sensor data is merged, aggregated, and cleansed. We assume that a DSMS is responsible for receiving the data from the many sensors .In addition, normal DSMS queries may be processed against this data.
- **Level 2 – Model**: A model at this level actually summarizes the data streams processed at level 1. This summarization is not like that used by aggregation in data warehouses, but rather a high level view of what the stream data has looked like and currently looks like. The model created is for all of the streams processed at Level 1. The model captures not only the data obtained by the sensors, but also the spatial aspect of the data (where the sensor is located) and the temporal aspect. It is important to note that the temporal aspect is not only the timestamp of the data, but also the ordering of the data from the sensors. In our overview we assume a dynamic first

order Markov chain is used. This dynamic model not only summarizes the data, but also captures concept drifts. Machine learning techniques allow the learning and forgetting of data over time. Clustering techniques are used to ensure a sub-linear growth rate of this model.

- **Level 3 – Domain Expert**: We call this level domain expert as we assume the primary users at this level are domain experts who need to examine the sensor data at an extremely high level. They rely upon the output of data mining applications applied to the model data. They will examine visual summaries of the data, output of anomaly detection software, and other data mining output. The exact view that each domain has depends completely on his needs. No real data exists at this level. The external view is only visual.

The proposed stream hierarchy can be compared to the more conventional data hierarchies/abstractions seen with memory hierarchies, Data Base Management Systems, and Data Warehouses. Table 1 summarizes these observations.

Of the four compared types of data abstractions, the memory hierarchy is the only one where higher levels of the hierarchy contain subsets of data at the lower levels. Moving from Level 0 to Level n, the amount of data decreases as does the access time. Data is migrated from lower to higher levels based upon explicit computer operations being executed or using a pre-fetch based upon predicted anticipation of access and locality.

Traditional DBMS data abstraction looks at three levels of data (external, conceptual, and physical). As opposed to the memory hierarchy, each level views the data differently. At the physical level, the data is stored and viewed as it has to be physically accessed. The conceptual view is how the DBMS views it-perhaps as a set of relations. The external view is how users see their data. Similar to the memory hierarchy, data is migrated up based upon its actual accesses. Sometimes the external/logical views are materialized to avoid slow access time.

The stream hierarchy can be viewed in a similar manner as data warehouses. As with a data warehouse summary of the operational data, the higher levels are summarizations of the actual stream data. However, type of aggregation performed at the higher may be quite different. Similarly, Roll-up and Drill-down types of operations can be used to find out detail information about the detail stream data which caused the observable behavior at the higher level. However, due to size limitations, the actual detail data may not be completely materialized. A cube view of the entire data, however, is probably not feasible due the infinite number of temporal points possible. Higher levels of cubes at temporal aggregate levels, (month, day, year) would, however, be possible. A major difference between our proposed hierarchy and a warehouse is that the data at the highest level is pushed rather than pulled.

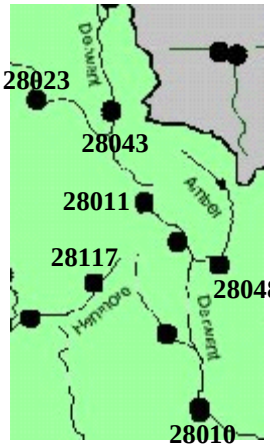


Fig. 2. Derwent Catchment Sensor Locations

3. STREAM HIERARCHY EXAMPLE

We illustrate our proposed hierarchy by examining real sensor data obtained from a set of six river sensors located in the Derwent River catchments as shown in Figure 2.

3.1 Level 0 - Physical Level

These six sensor data streams were provided by the British National River Flow Archive (http://www.nercwallingford.ac.uk/ih/nrfa/river_flow_data/index.htm). This data represents 1919 data points from November 1, 1971 to January 31, 1977 with a time interval of 24 hours between sample readings. We represent the data at each time point as a vector with six values, one from each sensor: <r1, r2, r3, r4, r5, r6>. Data in these time series represent cubic meters per second flow of the river at that sensor location.

3.2 Level 1 – DSMS

The Drewent river data is not actually accessed via a DBMS. However, the individual sensor readings are aggregated. The data on the aforementioned Web site provides the mean river flow averaged over all individual river flow readings for a 24 hour period. No other DSMS functionality is provided.

Table 1. **Comparison of Stream Hierarchy to Conventional Abstractions/Hierarchies**

	Levels	Lowest Level	Highest Level Abstraction	Inter-level Data Migration
Memory Hierarchy	n	External Storage	Subset/Cache/Buffer	Fetch/Prefetch
DBMS Data Hierarchy	3	Physical Storage	External View	Fetch, Prefetch
Data Warehouse	n	Operational Data	Cube/Multidimensional View	Aggregation
Stream Hierarchy	4	Sensor Data	Visualization/Triggers	Automatic Push

There have been several DSMS projects: STREAM (developed at Stanford) supports SQL-like queries. Aurora (developed at MIT, Brandeis, and Brown) supports processing of sensor data in a real time manner. Cougar (developed at Cornell) supports a distributed sensor network, and TelegraphCQ (UC Berkeley) [JA06]. Functionality of these systems varies but typically includes filtering/cleansing of sensor data, aggregation of sensor data, use of approximate and continuous queries, stream clustering, and need to adapt to changes in data behavior.

3.3 Level 2 – Model

The model level is the level at which data is permanently stored. However, this data is a summarization of the actually physical sensor data. Typical modeling of stream data often involves the use of Hidden Markov models, time series models, and sliding windows. Another technique which has been proposed to model stream data is that of "adaptive clustering" [JA06].

Modeling should include a technique to capture the temporal aspect of the stream, achieve a nonlinear growth rate via a technique such as clustering, and be able to both learn and forget data as the incoming model changes. Another popular stream modeling technique is that of a synopsis or summary of the data.

Although Level 2 could use any modeling technique, the only one that we are aware of that statisfies the temporal, continuous learning and clustering requirements for summarization is the *Extensible Markov Model (EMM)* [HMD04].

Definition 1. ***Extensible Markov Model (EMM).*** EMM is a dynamic first order Markov chain. Both the structure (number of nodes, number of arcs, and connections) as well as their labels are learned as the stream data arrives. In addition, nodes/arcs can be removed as the content of the stream shifts. The nodes in the graph are clusters of real world states where each real world state is a vector of sensor values. Any clustering algorithm can be used.

The nodes in the graph are labeled with the count of times that cluster has appeared as well as a representative (centroid/medoid) of the states in the cluster. Arcs in the EMM are labeled with the probability (based on ratio of number of times that arc as been used over the number of times the node at the head of the arc has appeared). As sensor vectors arrive into the DSMS, the EMM graph is updated to reflect the new counts. EMMs have been used to predict future values, identify rare events, and intrusion detection. They have been applied to river sensor data, automobile sensor data, VoIP traffic data, as well as other spatio-temporal data. We have previously reported on the performance of EMMs for flood prediction as compared with neural network and other Markov chain variants [DALBH05]. This work showed the superiority of the EMM approach based on both the error (using two different metrics) and the size.

Figure 3 (modified from [*HMD04*]) shows a stream of sensor data from six sensors and the corresponding construction of EMM. This hypothetical data comes from six sensors at seven time points. We assume that the DSMS has merged and cleansed the sensor data into vectors for each time point. In Figure 3 b) the creation of the EMM is shown at the seven time points. The EMM starts with no content and at Time 1 an EMM with one node is created. At each subsequent time point either a new node is created or the vector for that time point is added to a cluster represented by an existing node in the EMM.

3.4 Level 3 – Domain Expert

No data actually exists at the Domain Expert Level. Instead visualization of the stream data and results of triggers and lookmarks are pushed to the user. A trigger may be used to notify the expert of anomalous behavior/rare events. *Lookmarks*, on the other hand, are events that are predefined as events of interest in the stream data.

As with Level 2, we assume that any visualization technique may be used. There has been much recent work looking at visualization techniques for stream data [ABMGLP01]. Line graphs have been traditionally used to visualize time series data. A major problem with traditional line graph is that when outputs from many sensors are to be seen together either multiple graphs are created or all graphs are placed on one set of axes. In either case, the resulting

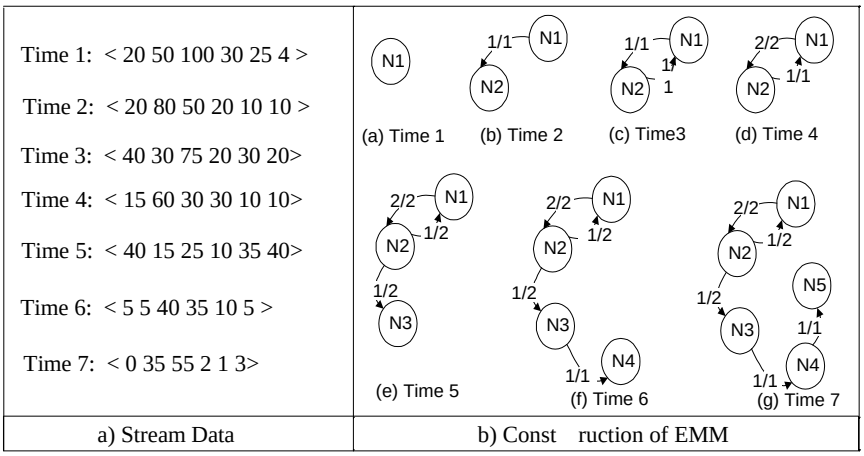


Fig. 3. EMM Model for Stream Data (modified from [HDM04])

figure(s) are difficult to read. Heat maps are often used to examine things like temperature. However, these techniques completely ignore the temporal aspect of the data.

Thus as a minimum, a visualization technique for the Stream Hierarchy needs to include the temporal aspect and facilitate viewing of output from multiple sensors at the same time. We are aware of only one visualization technique which satisfies both of these features: *Temporal Chaos Game Representation (TCGR)* [DQWMW06] alternatively called a *Temporal Heat Map (THM).* The THM has its basis in CGRs used to visualize DNA/RNA sequences. Within the stream data context, a THM is defined as shown in Definition 2.

Definition 2. ***Temporal Heat Map (THM)*** is a visualization technique for streaming data derived from multiple sensors. It is a two dimensional structure similar to an infinite table. Each row of the table is associated with one sensor value. Each column of the table is associated with a point in time. Just as stream data is infinite, the number of vertical columns is infinite. Each cell within the THM is a color representation of the sensor value. Rows may be normalized separately or together. Although different color schemes may be used, assuming a normalization of 0-1: colors with normalized sensor readings between 0 and 0.5 range from zero to blue and colors for values from 0.5 to 1 range from blue to red.

Figure 4 shows part of a THM for Derwent Sensor Data. Imagine a domain expert sitting at a monitor with this stream visualization gradually moving across his screen. Knowing which rows represent which sensors, he can instantaneously "see" what is happening with the data. The arrows in this figure show the movement with earlier times on the left and later on the right. The associated sensor numbers for each row are (bottom to top): 28023, 28043, 28011, 28117, 28048, 28010. Notice that sensor 28023 is actually for the smaller Wye river feeding into the Derwent. Similarly, 28048 is for the smaller Amber River. In the THM visualization you can see that both of these rows are lighter and thus have a smaller rate of flow, The higher flow rates for 28011 which contains flow from both the upper Derwent and Wye is normally much larger than the two up-river gauges.

4. CONCLUSIONS AND FUTURE WORKS

Stream mining should be viewed as a hierarchy of data and functions. In this paper we have provided a very brief overview of our hierarchical model. The hierarchy requirements are summarized as:

- Except for the inter-level functionality requirements, each level functionality is independent of the others and may differ across different implementations.
- The model used must capture time and ordering of data, be able to both learn and forget, and use some variation of clustering.
- Visualization at the domain expert level must capture both time and ordering. It addition it should be able to be easily "read" for many sets of sensors.

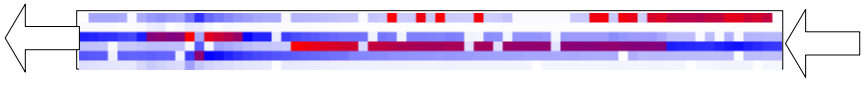


Fig. 4. Temporal Heat Map for Derwent Data

- Inter-level functionality must facilitate Drill-Down requests from the domain expert

The hierarchy described actually encompasses all prior work in each of the levels. That is, any type of sensors, their management, and deployment may be used. Any implementation of a DSMS may be used.

This represents a first step in analyzing this hierarchical data stream processing model. Much future work is needed to more formally examine the levels and functionality identified. Specific work to be performed would be description of inter-level functionality and perhaps language requirements.

We are excited about this new approach to sensor stream data management. It provides flexibility yet adaptability.

5. REFERENCES

[AAB05] M. H. Ali, W. G. Aref, R. Bose, A. K. Elmagarmid, A. Helal, I. Kamel, M. F. Mokbel, NILE-PDT: "A Phenomenon Detection and Tracking Framework for Data Stream Management Systems," *International Conference on Very Large Data Bases (VLDB)*, 1295 -1298, 2005.

[ABBCDIMSW04] Arvind Arasu, Brian Babcock, shivnath Babu, John Cieslewicz, Mayur Datar, Keith Ito, Rajeev Motwani, Utkarsh Srivastava, and Jennifer Widom, "STREAM: The Stanford Data Stream Management System," Dept of Computer Sc., Stanford Univ., 2004, http://infolab.stanford.edu/~widom/cs145/stream-paper.pdf.

[ACC03] Daniel J. Abadi, Don Carney, U˘gur C¸ etintemel, Mitch Cherniack, Christian Convey, Sangdon Lee, Michael Stonebraker, Nesime Tatbul, Stan Zdonik, "Aurora: a new model and architecture for data stream management," *The VLDB Journal (2003) / Digital Object Identifier (DOI) 10.1007/s00778-003-0095-z*

[BBDMW02] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and Issues in Data Streams," *Proceedings of the ACM Symposium on Principles of Database Systems*, 2002, pp 633-634.

[CFGR92] P. Cowley, B. Farrell, P. Gilna, and L. Russel, "Scientific data management:Real-world issues and requirements,". *SIGMOD* Panel and Abstract, June 1992.

[DGRTY03] Alan Demers, Johannes Gehrke, Rajmohan Rajaraman, Niki Trigoni and Yong Yao, „The Cougar Project: A Work-In-Progress Report," SIGMOD 2003, http://www.cs.cornell. edu/johannes/papers/2003/SigmodRecord2003-Sensor.pdf.

[D92] J. Dozier, "Access to data in NASA's Earth Observing System," SIGMOD Keynote Address and Abstract, June 1992.

[DALB05] Margaret H. Dunham, Nathaniel Ayewah, Zhigang Li, Kathryn Bean, and Jie Huang, "Spatiotemporal Prediction Using Data Mining Tools," Chapter XI in *Spatial Databases: Technologies, Techniques and Trends, Yannis Manolopouos, Apostolos N. Papadopoulos and Michael Gr. Vassilakopoulos, Editors*, 2005, Idea Group Publishing, pp 251-271.

[DQWMW06] Margaret H. Dunham, Donya Quick, Yuhang Wang, Monnie McGee, Jim Waddle, "Visualization of DNA/RNA Structure using Temporal CGRs," to appear in *Proceedings of the IEEE 6th Symposium on Bioinformatics & Bioengineering (BIBE06)*, October 16-18, 2006, Washington D.C. ,pp 171-178

[DALBH05] Margaret H. Dunham, Nathaniel Ayewah, Zhigang Li, Kathryn Bean, and Jie Huang, "Spatiotemporal Prediction Using Data Mining Tools," Chapter XI in *Spatial Databases: Technologies, Techniques and Trends, Yannis Manolopouos, Apostolos N. Papadopoulos and Michael Gr. Vassilakopoulos, Editors*, 2005, Idea Group Publishing, pp 251-271.

[ECHBB] Evans, J.M., Chang, T., Hong, T.H., Bostelman, R. and Bunch, W.R **"**Three Dimensional Data Capture in Indoor Environments for Autonomous Navigation," *NIST Internal Report #6912.*

[EGHK99] Deborah Estrin, Ramesh Govindan, John Heidemann, Satish Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks," In *Proceedings of the Fifth Annual International Conference on Mobile Computing and Networks* (MobiCOM '99), August 1999, Seattle, Washington.

[EGH00] D. Estrin, R. Govindan, and J. Heidemann, "Embedding the Internet", *Comm. of ACM*, Vol. 43, No. 4, May 2000, pp: 39-41.

[GO03] Lukasz Golab and M. Tamer Ozsu, "Issues in Data Stream Management," *SIGMOD Record*, Vol. 32, No. 2, Jun3, 2003, pp 5-14.

[GG06] Dina Goldin and Huayan Gao, "Dynamic Isoline Extraction for Visualization of Streaming Data," *Lecture Notes in Computer Sc.*, Vol. 3967/2006, pp: 415-426.

[HMD04]]Jie Huang, Yu Meng, and Margaret H. Dunham, "Extensible Markov Model," Proceedings IEEE ICDM Conference, November 2004, pp 371-374.

[HEB07] Georges Hebrail, "Data Stream Management and Mining," Tutorial at MMDSS 2007, Paris, September 10, 2007.

[HEI01] J. Heidemann *et al.*, "Building Efficient Wireless Sensor Networks with Low-Level Naming," *18th ACM Symposium on Operating Systems Principles*, October 21-24, 2001.

[HHM03] Joseph M. Hellerstein, Wei Hong, Samuel R. Madden, "The Sensor Spectrum: Technology, Trends, and Requirements," *SIGMOD Record*, Vol. 32, No. 4, Dec. 2003.

[HMA04] Moustafa A. Hammad, Mohamed F. Mokbel, Mohamed H. Ali, Walid G. Aref, Ann C. Catlin, Ahmed K. Elmagarmid, Mohamed Eltabakh, Mohamed G. Elfeky, Thanaa M. Ghanem, Robert Gwadera, Ihab F. Ilyas, Mirette Marzouk, Xiaopeng Xiong, "Nile: A Query Processing Engine for Data Streams," *IEEE International Conference on Data Engineering* (ICDE), 851, 2004.

[IGE00] C. Intanagonwiwat, R. Govindan and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *MobiCom* 2000.

[JA06] Ankur Jain, "Statistical Mining in Data Streams," Ph.D. Dissertation, University of California at Santa Barbara, December 2006.

[M03] Murli Mani, "Understanding the semantic of sensor data", *ACM SIGMOD Record*, Vol. 32 , Isssue 4, December 2003, pp: 28 – 34.

[MIC07]Motion Imaging Coorporation, "DMAS6 Motion Imaging and Analysis," October 2007, http://mi-as.com/wp-content/uploads/datasheets/dmas6.pdf .

[MFHH03] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. The design of an acquisitional query processor for sensor networks,". In *ACM SIGMOD*, Dec. 2003.

[NWS07] National Weather Service, "NOAA Implements New Water Resource Forecasts," Presentations available at http://www.nws.noaa.gov/oh/ahps/index.html, August 24, 2007.

[SFGJ93] M. Stonebreaker, Jim Frew, Ken Gardels, and Jeff Meredith , T. S. S. Benchmark. In *Proceedings of SIGMOD*, pages 2–11, June 1993.

[ZAI] Zaixian Xie, "Towards Exploratory Visualization of Multivariate Streaming Data". Computer Science Department, Worcester Polytechnic Institute, xiezx@cs.wpi.edu

Pushing Sensor Network Computation to the Edge while Enabling Inter-Network Operability and Securing Agents

Evens Jean, Yu Jiao, Ali R. Hurson, and Vijay Kumar

Abstract- **Sensor networks consist of small devices deployed in an area to perform a task through coordination and communication. The sensor nodes are generally equipped with one or more sensing devices and operate under severe resource constraints such as low-memory and low computational power. The latter constraint further restricts the types of applications that the deployed network is able to support. As thus, sensor networks are typically not deployed to support applications with real-time processing needs. Furthermore, while the network subsists in a dynamic environment, the tasks of the nodes are generally static and cannot adapt to changes in application requirements. As the tasks of nodes are static, multiple networks may need to be deployed in one area to support heterogeneous tasks even when these tasks have similar if not identical sensing requirements. To address these issues, this article proposes to push sensor network computation to the edge through the use of powerful and reconfigurable nodes as well as mobile agents. The proposed approach will enable networks to interoperate to allow reuse of existing deployed networks, prevent proliferation of networks with similar sensing capabilities and overlapping coverage area. Moreover, we herein discuss the expected benefits of our approach and address security threats facing the agent paradigm in traditional execution environments from both a centralized and distributed standpoint.**

Index Terms – mobile agents, security, distributed sensor networks,

I. INTRODUCTION

Sensor Network can be thought of as the resulting network that emerges from the, possibly random, deployment of multiple sensing devices, known as sensor nodes, in a particular area, to perform a task through coordination and communication. The sensor nodes are generally low-power, low-memory devices with highly constrained computational capability typically programmed prior to deployment with the ability to communicate with their peers, and equipped with some sensing apparatus [1]. Being generally cheap, the nodes may not be retrieved or reused in the event of failure. The data collected by the nodes are relayed to a special node in the network referred to as a base station or sink that does not suffer from the severe resource limitations noted earlier. The limitations present in Sensor Network led to the many challenges in the field but also to its wide acceptance in many application areas, especially due to the low-cost of sensor nodes.

Traditionally, once the sensor nodes are programmed and deployed, they coordinate with each other to accomplish one task and react to events only within the confines of what was predicted in the program. Various approaches have been proposed in the literature to circumvent this limitation. Such proposals have however not focused on bringing computation in sensor network to the edge to support application with real-time processing requirement, nor have they envisioned the need for different networks to interoperate towards the accomplishment of a task. Ultimately, sensor networks should be able to support a more diverse array of tasks, depending on the needs of applications or environmental stimuli. We here forth refer to any proposals that aim at allowing sensor networks to support various tasks dynamically as reconfigurable sensor network.

Within Sensor Network, the nodes' tasks are generally static although they subsist in a dynamic environment where adaptation is the key. Sensor Networks are not designed for interoperability, but rather for efficiency in terms of power. Multiple networks with varying sensing capabilities may cover a particular area especially when they are conducting disjoint tasks or are from different owners. In an urban setting, the proliferation of sensing nodes is not an attractive concept especially since it is not necessary. Generally, for every new application, a new network needs to be deployed. However, this can be circumvented in various cases if the intended coverage area of the new application has existing sensor nodes with the required sensing capabilities. By bringing sensor network computation to the edge and allowing interoperability of networks, we contend that the issue can be avoided altogether. The introduction of agents will further render the environment more flexible as the network can be made to

The National Science Foundation under the contract IIS-0324835 in part has supported this work.

E. Jean and A. R. Hurson are with the Department of Computer Science & Engineering at The Pennsylvania State University, University Park, PA 16801 (Email: jean@cse.psu.edu, hurson@cse.psu.edu)

Y. Jiao is with the Computational Sciences & Engineering Division of Oak Ridge National Laboratory, Oak Ridge, TN 37831 (Email: jiaoy@ornl.gov)

V. Kumar is with the School of Computer Science & Engineering.at the University of Missouri-Kansas City, Kansas City, MO 64110 (Email: kumarv@umkc.edu)

adapt to changing observations and, or requirements while intelligently processing collected data.

As an autonomous entity, dotted with intelligence, agents can migrate to the data sources of interest, intelligently process available information and make recommendations. The use of agents in information retrieval has been shown to be desirable [15]. One of the major hindrances to the widespread proliferation of agent application stems from their inherent ability to migrate, hence, as a consequence, a wide range of security issues have surfaced. Addressing such issues is a necessity in order to foster the use of the agent paradigm. Over the years, numerous platforms have been introduced to help foster the deployment of agent-based applications. However, such platforms have typically been restricted to traditional execution environments. With the recent advances in the field of Sensor Network, there is a growing interest in introducing the agent paradigm to the field particularly to render sensor network reactive to environmental stimuli.

To fully extract the benefits of agents in information retrieval, it is imperative that a secure environment be provided. The majority of the proposals introduced thus far to secure agent applications have generally been from a theoretical standpoint [4]. Moreover, such proposals view the issue from a centralized aspect attempting to protect either the agents or the hosts to which they migrate. The introduction of some of the proposed theoretical approaches to actual agent platforms is thus a necessity to secure the environment. Furthermore, it is of primordial importance that agent security be adequate to the environment in which the agents operate. By that we mean that agent security should not be studied solely from a centralized viewpoint but also from a distributed one.

This article intends to introduce a novel perspective on sensor network where nodes' computational power can be pushed to the edge, and heterogeneous networks can interoperate towards the accomplishment of task. Furthermore, the nodes in the network can be dynamically configured in response to environmental stimuli or based on the ever-changing needs of applications. Dynamic configuration of nodes will further allow the proposed architecture to evolve with changing protocols, as well as design and technological innovations. The expected contribution of the work is as follows:

- Interoperable and reconfigurable sensor network architecture with computationally powerful nodes able to support applications with real-time processing needs.
- Demonstrate the feasibility of the proposed architecture through vision-based tracking of targets resilient to environmental distractions, e.g., when the tracked object crosses path with others having similar or identical features.
- Address security issues of agents to foster their use in the environment and information retrieval in general.

In introducing the novel architecture, we will start by presenting the related materials in the following section. The proposed architecture is presented in section 3 followed by a discussion of our approach to agent security in section 4. We drew conclusions in section 5 highlighting the expected contribution of the work.

II. BACKGROUND

As we introduce our objective, it is of primordial importance that we ascertain that the reader has a thorough understanding of the main developments in the related fields of research. Hence we will start by introducing the current state of research in sensor networks in the next subsection followed by target tracking, agent security, and agent-based learning.

A. Sensor Network Reconfiguration and Interoperability

Once deployed, sensor nodes have a static task to accomplish and cannot adapt to changes in the environment that may require new parameters to be taken into account. A new set of nodes have to be redeployed should the need arise to conduct two tasks concurrently or alternately in an area covered by an existing network. There is thus a need for sensor networks to support dynamic programming of the nodes but also for the network to concurrently support multiple applications. One approach to addressing the aforementioned need in sensor network is to consider the sensor nodes as a set of data-stores into which queries can be injected to collect information that can be used by the sink to whatever purpose. Collecting information from the sensor nodes is inadequate in applications where the nodes need to interact with each other in order to reach a conclusion in real-time [6], as would be the case in distributed target tracking applications or any applications that require the use of distributed algorithms. The notion of "active sensors" was introduced in [6] to denote the path to addressing the reconfigurable need of sensor nodes through abstraction of the runtime environment. The "active sensors" approach typically makes use of virtual machines, script interpreters, and mobile agents to render sensor nodes' re-programmable and represents the strategy adopted by the following systems.

Maté aims at providing sensor networks with a flexible architecture upon which application specific scripting environments can be built [17]. User programs in the environment are generally short and simple as the virtual machine (VM) provides the functionalities common to specific application domains. This is an interesting approach to address the issue of reconfigurable sensor networks; however Maté suffers from assumption that the reprogramming, based on Trickle, occurs over all the nodes in the network and all of the nodes are coordinated for the execution of one specific application at any one point in time. Furthermore, Maté views the network as an isolated entity and does not address issues of interoperability with other networks.

SensorWare has also been introduced as an attempt to address the issue of reconfigurable sensor network by supporting the execution of mobile scripts [6]. SensorWare provides the necessary support allowing a Sensor Network to run multiple scripts simultaneously; as such, unlike Maté, it does not assume that the whole network is focused on only one task at any point in time. On the other hand, just as Maté,

it ignores issues of interoperability nor does it provide a service-based environment. The latest implementation of the system required 179KB of space with the core accounting for 30KB, which makes it very unsuitable for environments populated with nodes having very few storage capabilities.

Deluge has been designed to handle the dissemination of large data objects over Wireless Sensor Network (WSN) [9]. The Deluge protocol tackles the issue of network programming in sensor networks very efficiently. It takes into account the density of the network as well as the network connectivity. However the new protocol is limited by the fact that it currently assumes that all the nodes in the network need to be programmed and thus focused on propagating code updates to all nodes in the network. It is conceivable that an administrator may only want to update particular nodes in the network. Furthermore, it ignores issues of interoperability, and the need to support multiple tasks.

The need for the existence of reconfigurable sensor network has served as the main motivation for the development of Agilla [11], a mobile agent framework for sensor network. Agilla allows each node to support multiple agents that may or may not be cooperating to accomplish a task. Agilla allows for agents communication through tuple spaces. Agents in Agilla can clone themselves or move to another location carrying with them either their code and state or just their code. Agilla has been designed to allow more than one agent to execute on a node thereby tackling one of the issues that we hope to research. However, similarly to the previously discussed works, Agilla does not attempt to allow different networks to collaborate. In the event that neighboring networks all run Agilla, or the same agent platform for that matter, interoperation is still not possible as the individual networks might adopt different protocols. Moreover the system would not be able to adapt to changing protocols, a realistic occurrence as no standards have been established and the research field is open. Lastly, Agilla failed to tackle the issue of providing services to migrating agents.

Researchers at UC Davis have put forth a mobile agent framework built on top of the Mate virtual machine to allow the deployment of the programming paradigm within sensor network environment [21]. The framework allows agents to execute within an interpreter that attempts to prevent node crashing from corrupted agents. The interpreter implements the basic functionalities of agents, such as agent forwarding, so as to minimize the size of agent code that needs to be transferred from node to node. A case study for the use of agent in sensor network was conducted leading to the conclusion that the use of agents is particularly beneficial in the environment where only a subset of the network needs to be reprogrammed or the frequency at which nodes are reprogrammed increases [21]. The mobile agent framework for Sensor Network introduced by researchers at UC Davis suffers from the same limitations as Maté, including the lack of cross-network interoperability, with the exception that the entire network does not have to be reprogrammed. As an agent system, similar to Agilla, interoperability is possible under the assumption that the different networks support the same agent platform and the same set of protocols.

ActorNet is a mobile agent system for Wireless Sensor Network, which supports an asynchronous communication model, context-switching, multi-tasking, agent coordination as well as virtual memory [16]. The ActorNet platform is a virtual machine that can support multiple actors (agents) per node. The platform provides a unified environment for actors, which interact exclusively with the interpreter at the top level of ActorNet's architecture. ActorNet took a very efficient approach to the issue of reconfigurable sensor nodes, as did Agilla and the researchers at UC Davis, through the adoption of mobile agent to reprogram nodes. The platform, however, still does not allow interoperability of heterogeneous networks nor does it introduce a service-based approach to sensor network.

To sum up, ActorNet, along with the other proposals herein reviewed, have approached reconfigurable sensor network

Table 1: Comparison of the approaches to sensor network reconfiguration

System	*Platform*	*System Requirements*	*Heterogeneous Tasks*	*Node Multiplicity*	*Interoperability of networks*	*Service-oriented infrastructure*
Mate	Virtual Machine	TinyOS	No	Singular	No	No
SensorWare	Run-time environment	TinyOS	Yes	Multiple	No	No
Deluge	Network programming	TinyOS	No	Singular	No	No
Agilla	Agent system	TinyOS	Yes	Multiple	No	No
UC Davis framework	Agent system	Mate, TinyOS	Yes	Multiple	No	No
ActorNet	Agent system	TinyOS	Yes	Multiple	No	No

from a limited perspective in which the need to dynamically change the task of the network is the focus. We intend to stray from such a perspective by being concerned with not only the issue of changing the task of the network, but also that of allowing interoperability between various pre-existing networks. Moreover, provision of a service-based infrastructure where multiple applications, possibly from different individuals/owners, can utilize the network towards the accomplishment of disjoint tasks with varying sensing needs constitute another focus of this article. A table summarizing the features provided by the different approaches discussed is presented in table 1.

B. Target Tracking in Sensor Network

The use of Sensor Network in numerous applications, such as target tracking, has been the focus of numerous research efforts [1]. Through the implementation of target tracking applications, we intend to demonstrate the advantages of bringing sensor network computation to the edge and allowing interoperability between existing networks, while addressing some of the issues present in the field. As such, we herein present an overview of the recent research activities in the area.

Target tracking as the name suggests deals with the issue of following a particular object as it moves within an environment. Within the scope of Sensor Network, the environment is limited to any area where the nodes involved in the tracking are present. Schemes that have been put forth to allow sensor nodes to track a target efficiently were intended to minimize the power consumption of the network as a whole. Pattem et al [20] have identified four major approaches used in Sensor Network to track a target. The classification of the approaches is based upon the scheme used to minimize power usage in the network to track a target. Two of the identified approaches rely on the predicted future location of the target.

To deal with the randomness of a target's trajectory, Yang et al [24] introduced the Distributed Predictive Tracking (PDT) algorithm, which views the sensor network as a clustered system, where each cluster has a cluster head along with active sensors at the border. To track an object, the cluster head activates three sensors that can detect the target using their low sensing beam; if three such sensors cannot be found, the cluster head chooses three sensors that can detect the target using their high beam. The cluster head determines the next location of the target and informs the next appropriate cluster head of the ID of the tracked target. Mechitov et al [19] attempts to model the motion of the target through piecewise linear approximation and the introduction of a distributed algorithm, named Cooperative Tracking, to help in the prediction of the targets future location. The distributed tracking algorithm is based on four steps; during the first, each node measures the duration for which the tracked object is within its range. In the second step, the nodes exchange the recorded values for use in the third step to estimate the object's location by computing the weighted average of the detected values. Step four runs a line-fitting algorithm on the set of points calculated in the third step.

Furthering the trend of taking a distributed approach to tracking a target in Sensor Network, Zhang et al [26] introduced the Dynamic Convoy Tree-Based Collaboration (DCTC) algorithm. DCTC can be used to track moving targets in sensor network focusing on the issue of collecting and generating reports as the target is tracked. The proposed algorithm relies on nodes around the target collaborating with each other to determine a root. The root node would be in charge of predicting the path of the target and activating appropriate nodes along that path.

Limiting the environment to binary sensors, Aslam et al [3] deals with the issue of approximating a target's location by requiring nodes to relay to the sink one bit of information dictating whether the object of interest is moving closer or away from the node. This approach minimizes communication between nodes and the sink while allowing for the target's trajectory to be predicted by the sink. Taking an information-driven approach, Feng et al [10] proposes to inject queries into the area where the target is likely to appear initially. Once the query has reached a node where the event is likely to occur, the node upon detecting the event, determines the next best sensor to which to pass the information for tracking the event based on not only an estimate of future direction of the object but also on a minimized communication cost to transfer the information of the target to the next node.

Vision-based tracking aims at generating an object's trajectory through analysis of the objects location in successive frames of a video. Yilmaz et al. categorized the methods used to track objects as Point, Silhouette or Kernel based [25] depending on the representation of the object by the tracker. Regardless of how the tracked object is represented, occlusion, referred to as the obstruction of the tracked object by another tracked object or by the background, can occur. Common approaches to resolving occlusion includes location prediction, silhouette projection and even implicitly through generation of object tracks [25]. The issue herein referred to as path-crossing, has not received broad coverage and is concerned with determining the correct object to be tracked when the tracked object crosses path with another object with similar, if not identical features. The issue requires that the algorithm in use by the system reacts to such an event. Path-crossing introduces two or more objects in the system, with similar features in an approximate location thereby distorting any prediction of the tracked object's trajectory. The distortion occurs due to the fact that the motion of the objects in the system is random; hence determining which one of the objects whose paths have crossed was being tracked remains an open research issue to the best of our knowledge. Multi-view based tracking have been proposed in the literature, and is defined as a subset of kernel-based tracking [25]. Multi-view based tracking refers to the use of multiple cameras in determining the location of the tracked object. Our preliminary literature survey has not revealed any work exploring possible applications of multi-view tracking to path-crossing.

C. Mobile Agent Security

Mobile Agents refers to a programming paradigm focused around the ability for a program to halt its execution, move to a new environment where execution can then be resumed. In general, mobile agents are software entities that roam a network to carry out a task. These agents are typically mobile (though not a norm), autonomous, and perceivably intelligent. They can cooperate with one another to achieve a, not necessarily, common goal. The mobility of agents is not static; it can depend on current computation or be specified in advance by the user via an agent itinerary.

Mobile Agents find their applications in environments where there is a need to collect data from multiple sources over a network. The main advantage with the use of mobile agents is that it provides programmers with a new computational model that deviates from the traditional client-server approach. Agents take the computation to the data and as thus, as discussed in [15], can significantly improve the performance of a system. It is worth noting that generally mobile agents lend themselves nicely to searches and computation that requires parallel processing [15]. Over the years, numerous platforms have been released to support the development of mobile agents [2]. The full-scale deployment of the programming paradigm has always been hindered by the massive security threats that are inherent to its most attractive feature: its mobility.

Mobile agents are subject to security threats throughout their lifecycles that differs from those encountered in traditional client-server systems. The categorization of the threats plaguing mobile agents is done based on the origination of the attack; as such we have agent-to-host, as well as host-to-agent attacks. Such security issues in mobile agents have been studied [4] and some of the proposed solutions include but are not limited to the following:

- Code signing, access control; proof carrying code, and path histories to protect the hosts
- Tracing, obfuscation, trusted hardware as well as encrypted functions and data to protect the mobile agents.

Research in mobile agent security is still an open field, and many of these approaches remain theoretical at best.

D. Agent-based learning

Supervised learning focuses on the ability to extract patterns from a set of raw data whose categories are known. Various algorithms have been introduced to allow extraction of existing patterns in a data set. Such algorithms include Support Vector Machines (SVM), neural networks, decision trees, as well as boosting [12]. Boosting has an interesting property, in the fact that training occurs in stages. In each stage of boosting, a weak classifier is trained using a subset of the raw data. The set of trained classifiers yield the learning function used to determine how to categorize future data samples. Furthermore, due to the fact that boosting learning function emanates from several weak classifiers, it is easily adaptable to a distributed environment where each weak classifier may operate from different sources. It is this inherent ability of boosting that we hope to harness in this article to address the issue of agent security from a distributed standpoint through agent collaboration.

Agent collaboration has been the focus of various research efforts in recent years. Becker et al. studied the issue of confidence determination to ascertain its effect in collaborative agent systems [5]. The study showed that incorrect confidence-integration may propagate in a multi-agent system and thus change the collaborative answers of the agents. The problem was simplified by assuming that trust is not an issue between the collaborating agents. Within our approach in addressing distributed agent security, each of the collaborating agents is extremely flexible in integrating confidence factors to yield a collaborative result. Moreover, the agents do take trust into account in determining the dependence of their results upon other agents in the system as they collaborate to provide distributed security.

Chen et al. [7] presented a boosting-based hierarchical learning algorithm for experience classification. The work was motivated by the need for agents within a team to collaborate and learn from their past experiences, which may differ from one agent to another, as individual agents may only have a partial view of the team's environment. This learning algorithm attempts to take advantage of boosting by building a hierarchical framework where agents at the lowest level may only have a partial view of the system. Agents at the lowest level are trained using decision stumps based only on the feature set available to them. Agents higher up in the hierarchy are trained, not based on their observations, but using the classification results of the corresponding agents one level down in the hierarchy. Training in the proposed system is hierarchical and tightly coupled amongst agents as the classifiers are inter-dependent. The hierarchical learning system is not suitable to address security concerns as the system is built upon the assumptions that the agents are members of the same team, thus ignoring any trust issues. Furthermore, the fact that the system is built in a hierarchical fashion means that the final decision must originate from the root of the structure if it is to take into account the experience of every possible agent involved.

III. BRINGING SENSOR NETWORK COMPUTATION TO THE EDGE

In detailing our approach, we will structure the discussion based on the major areas that will constitute our focus. As thus, we will start by introducing our vision to enable interoperability and reconfiguration of the sensor network while pushing node's computation to the edge to allow support for time-constrained applications. Lastly, we will discuss our plan of implementing target tracking applications on the proposed platform resilient to path-crossing issues.

A. FPGA and Agent Based Reconfigurable and Interoperable Sensor Network

We have explored the need for sensor nodes to be made reconfigurable. Such needs stem not only from the future of Sensor Network requirements, but also from our vision of a time-shared environment where multiple overlapping

networks, each with its own set of sensing apparatus can be made interoperable to accomplish a task that could not be supported by any of the individual networks. Interoperability is particularly important in urban settings where proliferation of nodes in one area is not an option. With the diversity of sensing apparatus of the different networks, the overlay network can support a vast array of applications especially since it will not be subject to computational restrictions that have been imposed on traditional sensor nodes. Bringing computation to the edge, will allow the overlay network to support application with severe real-time processing constraints, such may be the case in vision-based target tracking applications. In vision-based target tracking, it is crucial that detection of objects occurs as quickly as possible and that knowledge of the object's current location is very accurate. What would be the point of using a tracker for on-line video surveillance if the information being processed occurred hours, even minutes ago? We have discussed numerous platforms aimed at supporting reprogrammable nodes in Sensor Network, through the use of virtual machines, script interpreters, or mobile agents. We have shown that such proposals generally do not support a service-based approach to sensor network. Systems such as Mate, assumes the network will only focus on one application at any point in time; therefore, the entire network is reprogrammed for any new application. While the agent-based systems can allow for some interoperability, it is only possible under restrictive assumptions such as the use of common communication protocols; moreover, it would further require that the networks share a common agent system. To address the limitations of the systems herein discussed, we propose to develop a new framework based on FPGAs and the use of mobile agents to achieve a time-shared Sensor Network environment that is interoperable, reconfigurable, with the ability to provide services to applications and strays away from the "dumb" node philosophy common to sensor networks.

Field Programmable Gate Arrays (FPGA) allows for a hardware chip to be reprogrammed in order to support a particular application. Our intent is to develop sensor nodes consisting of FPGAs; as such we expect our nodes to be pricier than traditional sensor nodes. Such an increase will refrain the use of the FPGA nodes in traditional Sensor Network environments, where nodes are assumed to be very cheap and thousands are easily affordable by an individual or company for a specific use. However, as an overlay network, with computationally powerful FPGA-based nodes, with great communication range, only a few number of such nodes need to be deployed (see Fig. 1). Being that the overlay network will consist of a small set of nodes, collection of such nodes in the event of failure or to replace their power source is feasible especially in urban settings. Their computational power will allow the nodes to react faster to environmental stimuli which will prove to be crucial to time sensitive applications. Moreover, as FPGAs are reconfigurable, the overlay network as a whole can be reconfigured to adapt to changes in the underlying networks such as the introduction of a new set of nodes or a new communication protocol. The re-programmability of nodes coupled with the speedup in execution are undeniable attractive features of the FPGA nodes. The exploitation of such features can easily occur with the coordination of multiple individuals or companies to deploy such nodes in conjoint areas of interest to share the network through deployment of mobile agents. Such agents would migrate to the FPGA nodes and harness the data available in a fashion that is optimal to the task at hand.

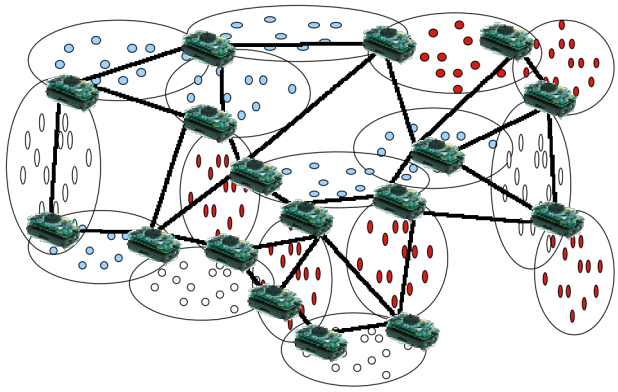


Fig. 1: Interoperable Overlay Network

As evidenced by our discussion thus far, the price of the nodes will affect the application domains suitable for our model. Being that the price of the nodes might serve as a deterrent for users whose applications would not benefit from the reprogrammable nodes; our model will thus be unfit for Sensor Network applications such as health monitoring. However, our model would be of great importance to military, urban, environmental and habitat monitoring applications as well as any Sensor Network applications that would benefit from reprogrammable nodes that can interoperate with existing networks. The size of the nodes further limits the application domain of our model; however, FPGAs are becoming increasingly smaller [22]. Presently, FPGAs are available in sizes comparable to the Berkeley motes, which are quite popular in the research community. Our model's limitations to specific application domains due to the size of the nodes will vary with the size of available FPGAs.

B. Target Tracking

Once we have concluded the implementation of our FPGA-based sensor nodes, we intend to proceed to the introduction of a target tracking application based on our design. As such the application will be agent-based and will attempt to specifically address the path-crossing issue discussed earlier. We will attempt to demonstrate that the issue is non-existent in a cooperative distributed approach of tracking a target, as we will claim that two objects are never crossing from different angles at the same point in time. As such, coordinating agents should be able to easily recover when a tracked object crosses path with another object with identical features. We will also study deployment strategies based on predicted location of the target for the agents involved in the tracking. We envision our target-tracking study to find its application in an area such as an airport where a person can be

tracked for security purposes. Moreover, a slight modification to our application can lead to a search-and-track application, where agents can be inserted in the system to locate an object or person and monitor the target's motion. Consider the case, in which a parent inadvertently loses track of his/her child in an airport for example, agents could be deployed to not only locate the lost child but also dynamically direct security or the parent to the child.

In retrospect, our proposal aims at tackling various existing issues in a set of diverse fields of study ranging from sensor network, to target tracking. A summary of what we hope to accomplish follows, highlighting the expected contribution of each undertaking:

- Overlay network to serve as a bridge and allow interoperable and reconfigurable sensor network environment
- Mobile agent framework to support the harnessing of data in the overlay network based on the needs of applications.
- Target tracking resilient to path-crossing issues as demonstration of the feasibility of this proposal

IV. MOBILE AGENT SECURITY

As applications in our proposal will be agent-based, it is imperative that we address the expected security concerns facing the paradigm. We will thus study agent security by focusing on one of the available agent platforms. Aglet is one of the numerous platforms introduced to support agent development; a pictorial representation of the platform's execution environment is provided in Fig. 2. Aglet is a library written in Java, released by IBM to support the development of mobile code; the platform is fairly documented, easy to install, has received great press coverage, and is currently maintained by the open-source community [13]. Furthermore, numerous application prototypes have already been introduced on the platform [13, 15], thereby making it a suitable choice in analyzing the security infrastructure in place to support secure agent applications. We have discussed the general classification of security threats to agent systems as being agent-to-host or host-to-agent. As thus, our intent is to analyze the security framework in place in Tahiti, the Aglet server, to ascertain that both entities are properly secured. Our study has revealed the following vulnerabilities in the Aglet platform:

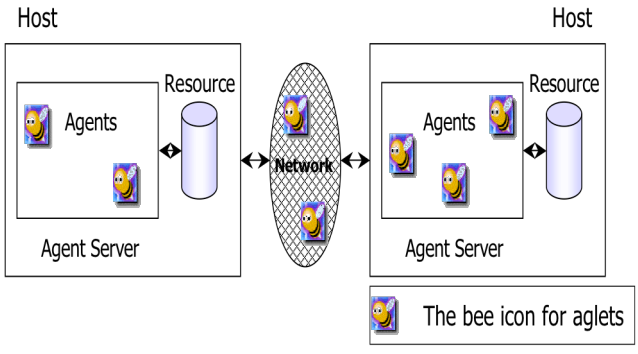


Fig. 2: Aglet Execution Environment

- Communication vulnerability established through the use of the Dsniff package to intercept agents as they are being transmitted from one host to another. This has led us to conclude that the Aglet framework cannot satisfy the requirement of agents to migrate exclusively to intended hosts.
- Data vulnerability has not been addressed in the Aglet framework. While there is no acclaimed solution to the issue, it is imperative that users be able to determine if their data has been tampered with, and determine the malicious host
- Resource vulnerability determined through analysis of the lifecycle of agents in the environment. Through seemingly normal transition of states, agents can wreak havoc in hosts through repeated state transitions such as cloning.

Details on the aforementioned studies can be found in [13]. Based on the results of our experiments, we have thus undertaken the task of addressing agent security which resulted into the introduction of a new server namely Secure Aglet Server (SAS). The following subsections detail our contribution thus far to address agent security from both a centralized and a distributed standpoint.

A. Centralized Agent Security

The Communication Layer of Aglets makes use of an unsecured protocol, thus leaving the framework open to a range of attacks, which we intend to address in our research. The current industry standard in addressing communication security being SSL, keeping in line with our intent of fostering the adoption of agents in commercial applications, we opted to implement SSL in Tahiti, using the Java Secure Socket Extension (JSSE). Through the use of SSL sockets, we have endowed the Aglet framework with the ability to communicate over secure channels capable of authenticating the parties involved, refusing unsecured connections and adjusting the security level on the channels. Through the availability of SSL in SAS, administrators will gain control of the level of security enforced on network links; most importantly, it provides a standard solution trusted in the industry to handle secure communication.

Granting Aglets the ability to detect tampering with their data required extended functionalities of the server. The Runtime Layer of SAS is extended to support the creation of Message Digests using the Java Cryptography Extension (JCE) as well as the ability to digitally sign objects. We provide Aglets with a java class library that implements the concept of Read-Only data. With the new functionalities of SAS in place, the library obtains a signed copy of the message digest computed by the host along with the host's certificate. An Aglet can retrieve the message digests stored by the library and use the corresponding certificate to ensure that its data has not been tampered with. The introduction of computed message digests and digital signatures in the Runtime Layer of SAS provides Aglets with the capability of detecting active malicious hosts in the agent's itinerary.

Dealing with the possibility of an Aglet overusing the resources of a host through seemingly normal transition between its lifecycle states (see Fig. 3), requires the design of a scheme to not only specify and track the resources in use by an Aglet but also to take proper actions once an Aglet attempts to overuse the host's resources. The design of such a scheme led us to the introduction of a MonitorAglet in SAS. The MonitorAglet tracks the number of instances of an Aglet, based on the Aglet's properties such as the corresponding ID, to ensure that the specified limit is never exceeded. Within the scope of SAS, we define instances as the instantiation of an Aglet or Message object. Furthermore, an Aglet *B* is an instance of an Aglet *A* if and only if one of the following is true

- *B* belongs to the same resource object as *A*
- *A* has created, retracted, or activated *B*
- *B* is a clone of *A*.

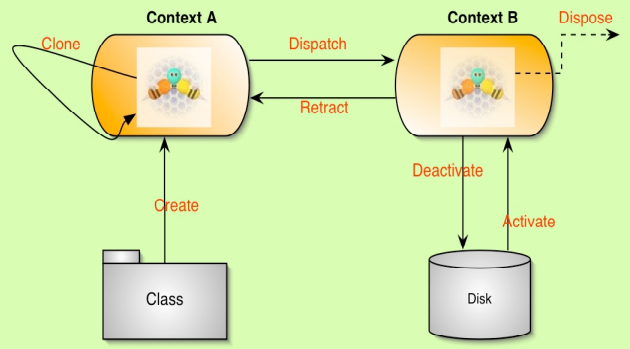


Fig. 3: Lifecycle states of Aglets

Once an Aglet has reached its instance limit, the MonitorAglet prevents the creation, activation, or retraction of any other instances of the Aglets in the system until one of the instances has been deactivated, dispatched, or disposed of. As we attempt to protect hosts against malicious agents, we have introduced powerful capabilities to the Aglet framework. It is now possible to limit the number of instances of an Aglet executing on a host, thereby preventing the attack described earlier.

In addressing centralized agent security, we have introduced a new agent server, namely SAS, endowed with secured communication, ability to detect tampering of agent's data along with prevention of over-usage of host's resources. The interested reader is referred to [13] for further details on the aforementioned schemes.

B. Distributed Agent Security

The introduction of the following security scheme stems from the realization that agents interact in a distributed environment; hence, similarly, agent security needs to be validated in a distributed manner. As hosts monitor agents, data regarding the actions of the agent can be recorded. Our work is based on the assumption that there is a relationship, though not clearly defined, between the actions of an agent and the intent of such agent; whether the intent is malicious or not. The definition of the set of actions that can help determine whether an agent is malicious will vary from one host to another and such actions are herein referred to as threatening actions. The consistent fact will remain however, that a malicious agent on one host is highly likely to represent a threat to the security of future hosts. Due to the variation in what constitutes a malicious agent, the proposed scheme relies on supervised learning thereby allowing flexibility in identifying potential threats.

Our approach in tackling the problem is through the introduction of a variation of the Boosting-learning algorithm, here forth referred to as DASAC. In order to determine whether an agent is malicious, DASAC relies on collaboration between the current host and past hosts visited by the agent. The current host acts as a decision maker (DM); every hosts including the current one act as base learners. We attain the required flexibility by allowing each host in the system, as base learners, to be trained independently and based on different feature sets. The base learners are trained as follows:

- Implement a binary classifier, which can be a decision tree or any other classifier, where 1 is the class of malicious agent and -1 otherwise.
- Train the classifier using a sample data set with the threatening actions of the host as the various features of each training instance.

Note that each host in the system may serve as a base learner and as a decision maker depending upon its contribution to the current decision-making process. The base learners, being trained independently, may implement various classifiers depending on the host's administrator. Furthermore, the feature set used by each hosts may differ, a discussion on what feature set could be used is provided in [14].

Within DASAC, classification of an agent by the decision maker is based on the following steps:

- If the host has had prior experience with the agent, the base learner of the host is used to classify the agent; else, the agent is assigned to the default class of 0.
- Every host in the agent's history are contacted and asked to communicate to the decision maker their classification of the agent as determined by their respective base learners
- Using the possibly diverse experiences of other hosts, the decision maker determines whether to allow an agent to execute or not.

In essence, a DM forms a hierarchical structure with the various base learners of the hosts in the distributed environment to thwart attacks. In the final steps, a DM could use various techniques to reach a consensus such as majority-vote. We however recommend a version of weighted sum tailored to the problem at hand as specified in (1), where Ψi represents the class to which an agent has been assigned by the base learner of a host. We allow Ψi to possibly have a value of 0 in order to ignore a base learner that does not have any information on the agent as such may be the case for the learner on the current host. Furthermore, τi, and λi represent, respectively, the trust, and confidence levels associated with each host being contacted.

$$x = \begin{cases} 1 & \forall \sum_{i=0}^{n} \tau_i * \lambda_i * \Psi_i > 0 \\ -1 & \forall \sum_{i=0}^{n} \tau_i * \lambda_i * \Psi_i < 0 \\ 0 & Otherwise \end{cases} \quad (1)$$

The recommended version of majority vote stems from our observations of the underlying mechanisms in inter-human collaboration. Consider the case where a person, *A*, asks a friend, *B*, for his/her opinion on a puzzling question; *A* does not blindly believe *B*'s assertion. Instead, *A* weighs his/her opinion and confidence on the topic with *B*'s recommendation based on two factors; namely, how much does *A* trust *B* and how confident is *B* in his/her assertion. Thus, the confidence level, in the proposed majority vote scheme, is determined by the accuracy of the classifier used in a host and varies between 0 and 100. The confidence of a host is communicated to the DM along with the classification of an agent. The trust level, on the other hand, can be defined statically by the system administrator of a host based on the reputation of a particular remote host. As the definition of trust levels does not carry over from one host to another, administrators are free in setting the limits of trust values in their systems.

If an agent is allowed to execute in the system, the decision maker keeps track of the actions of the agent. It can then periodically attempt to classify the agent and thus adapt to agents that may execute malicious code only on specific hosts. The frequency upon which to re-classify an agent is left as an implementation detail as it will vary upon the requirements of a host.

$$\Delta = n * \left[\max(\tau_i) * \max(\lambda_i) * \max(\Psi_i) \right]$$

$$SL = \left\lceil \beta * \frac{\sum_{i=0}^{n} \tau_i * \lambda_i * \Psi_i}{\Delta} \right\rceil \quad (2)$$

To provide administrators with hands on control on whether or not an agent should be allowed to continue or start execution, we introduced the notion of Security Levels (SL) of agents. The SL of an agent is defined in (2) as the ceiling of the product of its weighted-sum, as computed in Equation 1, and the number of security levels in the system (ß). The result is divided by Δ, representing the maximum sum of products multiplied by the number of cooperating hosts. Note that Δ is always greater than 0 as *n* takes into account the current host as well.

While the SL could be calculated for all possible value of the weighted-sum, one should note that it is not of importance when the weighted-sum is 0 or less as such agents have not been classified as malicious. Using the SL, the system can be made to be semi-autonomous, requiring human assistance once a threshold has been reached. Agent-human interaction can further increase the efficiency of the system as the agent can be made to adjust its classifier based on such interactions. Thus, DASAC may decide to use the collected data about an agent, classify it based on its interaction with an administrator and inserts the information in the pool of training data. The classifier can be periodically retrained thereby leading to an adaptive security system. Details regarding DASAC, its implementation in SAS along with experimental results are provided in [14]; the interested reader is referred to the work for a deeper understanding.

Addressing distributed agent security has resulted in the introduction of DASAC harnessing the ability of agents to learn and collaborate in order to protect hosts in the environment. The introduced framework builds on the idea of boosting to allow host protection from malicious agents based on the agent's reputation in the system. Moreover, the framework introduces the notion of security levels to allow human-agent interaction in rendering a robust and flexible security system.

V. CONCLUSION

Completion of the herein discussed work is expected to produce a new FPGA-based mobile agent platform for reconfigurable sensor network. The introduced platform will allow for pre-existing network to interoperate towards the accomplishment of novel tasks. As the introduced nodes will be more powerful, and provide support for mobile agents, the system will be apt in supporting distributed applications with real-time processing constraints. Furthermore, by bringing computation to the edge and allowing interoperability, the concept of services can be available on the nodes to be harnessed by agents based on their needs. The availability of services will reduce the size of migrating agents, while the overlay network will reduce relayed communication from one node to the next. Our test-case implementation of a mobile agent target tracking application for the proposed work will allow us to study the path-crossing issue discussed earlier. Furthermore, we have already secured an agent platform, namely Aglet, to allow secure deployment of agent applications. The secured platform addresses agent security from both a centralized and distributed standpoint.

Our main contribution to the field will include not only the introduction of an interoperable and reconfigurable agent-based platform; but also we will have addressed the path-crossing issues existing in target-tracking applications. To the best of our knowledge, our work will represent the first such work both in terms of our approach to reconfigurable sensor

network and to the avoidance of losing a tracked target as it crosses path with another object with similar features.

REFERENCES

[1] Akyildiz I. F, Su W., Sankarasubramaniam Y., Cayirci E. “A Survey on Sensor Networks” In IEEE Communications Magazine August 2002.

[2] Altmann J. Gruber F., Klug L., Stockner W., Weippl E. “Using Mobile Agents in Real World: A Survey and Evaluation of Agent Platforms” In 2nd Workshop on Infrastructure for Agents, MAS and Scalable MAS at Autonomous Agents, Montreal, Canada, 2 May 2001

[3] Aslam J., et al.“Tracking a Moving Object with a Binary Sensor Network” In Proceedings of ACM Sensys 2003, Los Angeles, California, 2003.

[4] Bierman E., Cloete E. “Classification of Malicious Host Threats in Mobile Agent Computing.” In Proceedings of SAICSIT, 2002. Pages 141-148

[5] Becker R., Corkill D. D., “Determining Confidence When Integrating Contributions from Multiple Agents” In AAMAS 2007, Honolulu, Hawaii, May 2007

[6] Boulis A., Han C., Srivastava M. B. “Design and Implementation of a Framework for Efficient and Programmable Sensor Networks” In Proceedings of the 1st. International. Conference on Mobile Systems, Applications and Services, pages 187-200, New York, NY, USA, 2003. ACM Press.

[7] Chen P.-C., Fan X., Zhu S., Yen J. “Boosting-based learning agents for experience classification” In Proceedings of the 2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, p. 385-388, 2006.

[8] Collins R., Lipton A., Fujiyoshi H., Kanade T. “Algorithms for cooperative multisensor surveillance”. In Proceedings of IEEE Vol. 89, No. 10, Pages 1456-1477

[9] Dutta P. K., Hui J. W., Chu D. C., Culler D. E. “Securing the Deluge Network Programming System”. In IPSN 2006, Nashville, TN

[10] Feng Z., Jaewon S., James R. “Information-Driven Dynamic Sensor Collaboration for Target Tracking” IEEE Signal Processing Magazine, 19(2), Mar 2002.

[11] Fok C., Roman G., Lu C. “Rapid Development and Flexible Deployment of Adaptive Wireless Sensor Network Applications” In Proceedings of ICDCS 2005, Columbus, Ohio, June 6-10, 2005, Pages 653-662.

[12] Hastie T., Tibshirani R., Friedman J. H. The Elements of Statistical Learning. Springer, 2001

[13] Jean E., Jiao Y., Hurson A.R., Potok T.E. "SAS: A secure aglet server," In Proc. of Computer Security Conference 2007,

[14] Jean E., Jiao Y., Hurson A.R., Potok T.E. Boosting-based Distributed Adaptive Security-Monitoring through Agent Collaboration, In Second International Workshop on Agent and Data Mining Interaction ADMI 2007

[15] Jiao Y., Hurson A. R. “Performance Analysis of Mobile Agents in Mobile Distributed Information Retrieval Systems – A Quantitative Case Study”. Journal of Interconnection Networks, Vol. 5, No. 3, Pages 351-372, Special Issue, 2004.

[16] Kwon Y., Sundresh S., Mechitov K., Agha G. “ActorNet: An Actor Platform for Wireless Sensor Networks” In. AAMAS 2006, Hakodate, Japan

[17] Levis P., Gay D., Culler D. E. “Bridging the Gap: Programming Sensor Networks with Application Specific Virtual Machines” UCB//CSD-04-1343. August 2005.

[18] Lotfinezhad M., Liang B. “Energy Efficient Clustering in Sensor Networks with Mobile Agents” in Proc. IEEE WCNC'05, Mar. 2005, Pages 1872--1877.

[19] Mechitov K., Sundresh S., Kwon Y, Agha G. “Cooperative Tracking with Binary-Detection Sensor Networks” Technical Report UIUCDCS-R-2003-2379, University of Illinois at Urbana-Champaign, 2003.

[20] Pattem S., Poduri S., Krishnamachari B. “Energy-Quality Tradeoffs for target Tracking in Wireless Sensor Networks” In International Symposium on Aerospace/Defense sensing Simulation and Controls, Aerosense, April 2003.

[21] Szumel L., LeBrun J., Owens J. D. “Towards a Mobile Agent Framework for Sensor Networks”. In the Second IEEE Workshop on Embedded Networked Sensors, Sydney, Australia, Pages 79-87, 2005.

[22] Telikepalli A. “Power Vs Performance: The 90nm Inflection Point” http://www.xilinx.com/bvdocs/whitepapers/WP223.pdf White Paper, Xilinx April 25, 2005.

[23] Wang Q., Chen W., Zheng R., Lee K., Sha L. “Acoustic target tracking using tiny wireless sensor devices” In IPSN 2003. LNCS 2634, Pages 642-657, 2003

[24] Yang H., Sikdar B. “A Protocol for Tracking Mobile Targets using Sensor Networks” Proceedings of IEEE Workshop on Sensor Network Protocols and Applications, 2003.

[25] Yilmaz A., Javed O., Shah M. “Object tracking: A survey” In ACM Computing Surveys Vol. 38, No. 4, Article 13 (Dec. 2006)

[26] Zhang W., Cao G. “DCTC: Dynamic Convoy Tree-Based Collaboration for Target Tracking in Sensor Networks” IEEE Transactions on Wireless Communication, Vol. 3, No. 5, Pages 1689-1701, September, 2004.

Real Time Event Detection for Decision Support System using Wireless Sensor Networks

Veena K.N.
Senior Lecturer
Dept. of Telecommunication
JNN College of Engineering
. Shimoga, India-577204

Dr Vijaya Kumar B.P.
Professor and Head
Dept. of Computer Science
JNN College of Engineering.
Shimoga, India-577204

***Abstract*-Real time event detection is a common requirement in any time critical application like Human centric applications, Robotics, Precision Agriculture, Military and Medical applications etc. Sensing an event means to identify its time of occurrence, place and magnitude. Real time event detection is to sense and communicate the same to the Decision Supporting System (DSS) at the minimum possible time duration. Monitoring, identification of events requires a continuous sensing, processing and communicating the information either in a partial or fully distributed environment and is a challenging issue. In order to tackle the above challenge Wireless Sensor Networks (WSNs) provides a cost effective and efficient solution. In the proposed work the system is designed for real time event detection using WSNs. The system will continuously monitors and identifies the event of occurrence at any point of time, place and its magnitude. The detected events are processed and communicated through the sensor networks by taking shortest path and reliable nodes to reach the DSS. The proposed system is simulated by using TinyOS for WSN with necessary environmental, system and networking parameter for real time event detection. Simulation results are taken to evaluate the performance of the proposed system with respect to event sensing and its magnitude.**

I. INTRODUCTION

Event detection is well specified time bound along with the place and the magnitude of the event based on the situation in a given environment are the major issues in real time system design at the decision making point. The complexity of these issues still increases in distributed systems, for example medical, habitat monitoring, military, agriculture applications. The application providing such services requires the real time event detection for central Decision Support System (DSS).

Recently developed and well established Wireless Sensor Networks (WSNs) having a wide variety of applications and systems with various requirements and characteristics have shown effective results for event detection. The complexities of such network design for real time event detection have increased with respect to hardware and software. Integrating sensor nodes for event sensing and supporting network for information processing and communication infrastructures to form wireless sensor networks will have a significant impact on real time event detection.

Wireless Sensor Networks are usually a large number of sensor nodes, which are tiny, compact and low cost embedded devices, which can be readily deployed in various types of unstructured environments within predefined and specified area. Each node of the network consists of three components: a variety of sensors to acquire information about the observed space, a low range wireless transceivers and a computing system having highly resource constrained networking system. Efficiency of such systems would improve by providing distributed, localized and energy efficient techniques and a set of well-defined protocols. In this work we have proposed a system for real- time event detection for decision support system to monitor and detect the event of occurrence at any point of time, place and the region of interest. Simulation for the proposed model is carried out by using TinyOS to evaluate the performance of the system.

The organization of the paper is as follows. A brief discussion on some of the related works is explained in section II. Section III explains the concept of real time system. The description of the proposed system for real time event detection using Sensor Network simulated by TinyOS simulator (TOSSIM) is narrated in section IV. Simulation and results are described in section V and concluding remarks are given in section VI.

II. RELATED WORKS

In this section a brief discussion on the related works pertaining to real-time systems and its applications in WSNs are discussed. Wireless Sensor Networks for In-Home Healthcare [1] and its Potential and Challenges is presented. The data will be collected and reported automatically there by reducing the cost and inconvenience of regular visits to the physician. Real-Time Target Tracking Using Wireless Sensor Networks VigilNet [2] a large-scale sensor network system, which tracks, detects and classifies targets in a timely and energy efficient manner is proposed. The use of non-destructive bit-wise arbitration on the MAC layer [3] for real-time message ordering in wireless sensor networks is presented. In Sensicast [4] Wireless Sensor Network system is used for pharmaceutical and life sciences facilities which provide accurate monitoring of temperature, humidity, and other key data in their facilities is presented. RAP, a Real-Time Communication Architecture [5] for Large-Scale Wireless Sensor Networks, provides convenient high- level query and event services for distributed micro sensing applications performing real time monitoring and control. A Real-Time Communication Protocol for Sensor Networks are proposed in [6] for providing real time communication services. Real-time Power-Aware Routing Protocol [7] in Sensor Networks, which achieves application-specified communication delays at low energy cost by dynamically adapting transmission power and routing decisions, is presented. Wireless Sensor Networks for Monitoring of Large Public Buildings [8] can be deployed for monitoring the real time response of structures to strain and ambient

vibration, monitoring and controlling of indoor environment, and helping in extreme event response are given. Wireless Sensor Networks for Ecology [9] to facilitate the collection of diverse types of data (from temperature to imagery and sound) at frequent intervals—even multiple times per second—over large areas, allowing ecologists and field biologists to engage in intensive and expansive sampling and to unobtrusively collect new types of data is presented.

III. REAL TIME SYSTEMS

In Real time systems [10] timeliness is as important parameter as the correctness of the outputs. Such real time system design and analysis is a complex issue and requires an application specific time bound and varies drastically as per the user, system and resources involved in the application. The information accessed from the event of occurrence need to be processed at a regular and timely rate. For example an aircraft uses a sequence or stream of accelerometer pulses to determine its current position, in addition, other systems require a rapid response to events that are occurring at non regular rates, such as an over temperature failure in a nuclear plant. In some sense it is understood that these events require real time processing. The system logical correctness is based on both the correctness of the outputs and their timeliness and often is reactive or embedded systems. Reactive systems are those that have a continuous interaction with their environment. Example: a fire control system that constantly reacts to buttons pressed by a pilot. In designing such real time systems embedded hardware circuits are used to take real time decision at the DSS. Real time systems can be classified as soft real-time systems – that is, systems in which performance is degraded but not destroyed by failure to meet response time constraints. Hard real-time systems, where failure to meet response time constraints leads to system failures. The applications involving distributed environment with event detection time, the place of occurrence of the event and its magnitude are important parameter to be considered. The design and analysis of such system is a challenging issue.

IV. PROPOSED MODEL

In this section we have described a proposed a system for real time event detection for decision support system to monitor and detect the event of occurrence at any point of time, place and the region of interest. The information is processed and stored in the processor unit, which will have a minimum computation capability and could be able to communicate to a shorter distance range. Each processing unit has minimum resources like memory, battery power, and transceiver with short communication system. This tiny processing unit is spread over the region of interest. They form a self organized network and could be able to communicate by using simple communication protocol. A typical real time event detection system with sensor network is as shown in figure1. It consists of mote having sensing, processing, transceiver and power units. The physical environmental event data is accessed by the sensor present in the mote. This data is given to the analog to digital converter for converting the analog signal to digital signal. This digital data is given to the processor. The processor processes the data depending on the application requirement. The processed data is then transmitted over the transceiver to the sink. The sink then sends the data to the central control system or DSS.

The simulation model for the event detection using TinyOS is as shown in figure 2. In this proposed model the magnitude of the event is identified by the threshold fixed for each sensor node which could be controlled from the central control unit. End access event time and their magnitude can be controlled by communicating threshold values, values, synchronous and asynchronous monitoring commands to the sensor node at any critical region of interest. In this system each individual sensor node, processing unit and sink could be programmed as and when the decision making and the parameters need to be changed. For example, to identify different thresholds at any region, we can change the magnitude at which the sensor should identify the corresponding magnitude of the event. The region of interest can also be changed accordingly along

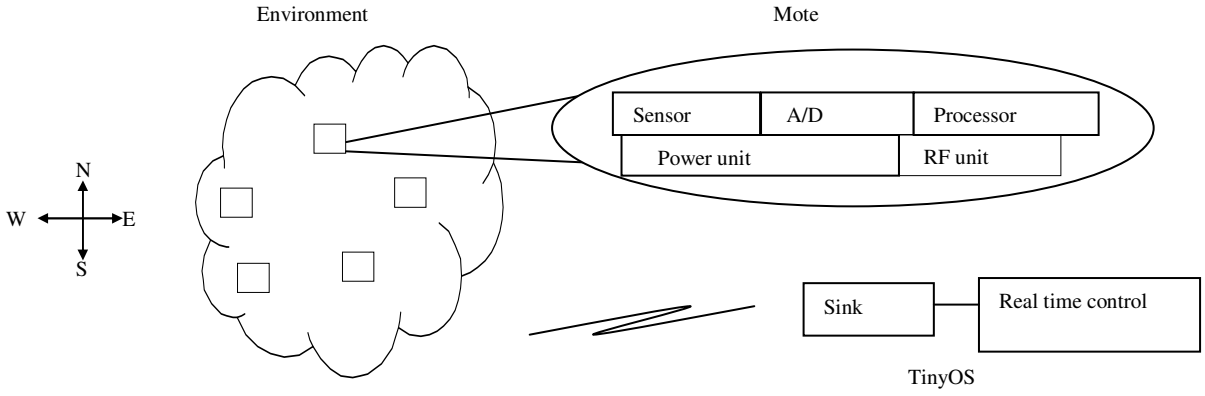


Figure 1 A typical real time sensor network model

with sampling rate, synchronous or asynchronous the proposed system should be able to control the parameter which needs to be monitored depending on the application and their event that is necessary for Decision Support System.

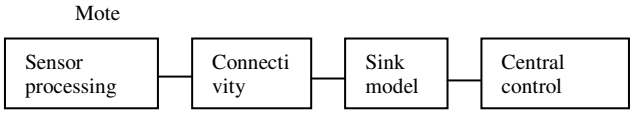


Figure 2 Simulation model using TinyOS

V. SIMULATION AND RESLULTS

The TinyOS simulator has been configured for the proposed system using components, interfaces, command and event handlers etc. and using available sensors. Each of the components are tuned to the parameter defined in the proposed model like different thresholds and different sensor nodes with proper interfacing among the sensor. The individual components are written using NesC. This proposed system has been tested to evaluate the performance in terms of real time event detection parameters like threshold values, event of occurrence and traffic flow in the system for synchronous and asynchronous monitoring. The available sensor on the TOSSIM is used to collect the data. Simulation experiments are carried out rigorously by taking different number of nodes in sensor networks. The simulation program is run for many iterations and for different sampling rates. The values sensed by the sensor for sampling rate 250ms is as shown in the figure 3.

The number of messages sensed by the mote for different sampling rates 250ms, 500ms and 750ms is as shown in the figure 4. From the result it shows that as the sampling rate increases the number of sensed messages also increased. Therefore the number of messages transmitted also increases. This increases the traffic transmission delay and processing time etc. But as the sampling rate increases the accuracy in reading the data also increases. We have set different values of thresholds at sensor node, before they start sending the event data. The number of messages sent for different values of thresholds for a particular value of sampling rate is shown in figure5. We can see that as the threshold value increases the messages transmitted is decreased, thus the traffic is also decreases in turn the transmission delay. Thus we can send the real time data to the end system i.e., DSS effectively. Also it is observed that from this the system could identify the event of occurrence, place and its magnitude for real time decision making.

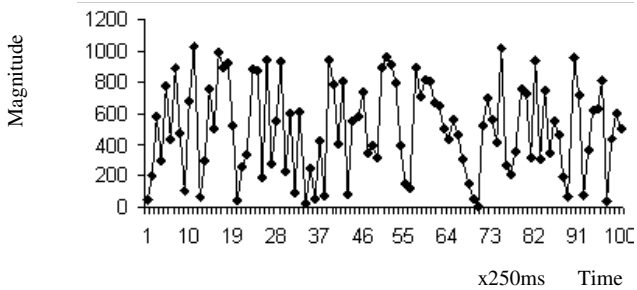


Figure 3 Sensed data

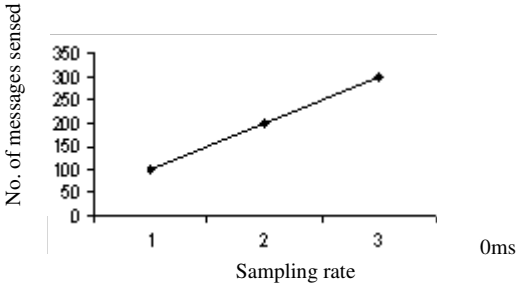


Figure 4 plot of no. of messages sent with respect to sampling rate

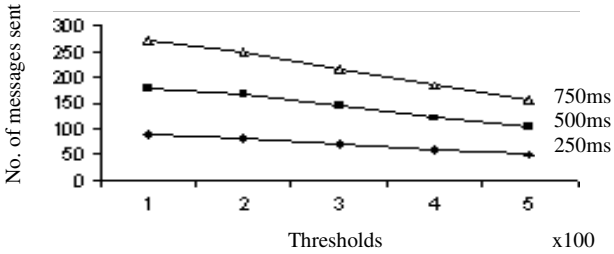


Figure 5 plot of no. of messages sent with respect to

VI. CONCLUSION

In this work we have proposed a system for Real time event detection for Decision Supporting System using WSN. The simulations are carried out to evaluate the performance of the proposed method with respect to different parameters of sensor networks and applications requirements. The proposed system is simulated by using TinyOS for WSN with necessary environmental, system, networking parameters for real time event detection. Simulation results are taken to evaluate the performance of the proposed system for event detection. This method of implementation would give scope for more understanding the real time behavior of system parameters of WSN and environmental parameters to enhance the operational efficiency.

REFERENCES

[1] J. Stankovic, Q. Cao, T. Doan, L. Fang, Z. He, R. Kiran, S. Lin, S. Son, R. Stoleru, and A. Wood.: Wireless Sensor Networks for In-Home Healthcare. In: Potential and Challenges, HCMDSS 2005, April 2005

[2] Tian He, Pascal Vicairey, Ting Yan, Liqian Luo, Lin Gu, Gang Zhou, Radu Stoleru, Qing Cao_, John A. Stankovic and Tarek Abdelzaher.: Achieving Real-Time Target Tracking Using Wireless Sensor Networks. In: ACM Transactions in Embedded Computing Systems, 2007

[3] Albert Krohn, Michael Beigl, Christian Decker and Tobias Zimmer.:TOMAC- Real-Time Message Ordering in Wireless Sensor Networks Using the MAC Layer. In: 2nd International Workshop on Networked Sensing Systems (INSS), San Diego, USA, June 27 - 28, 2005

[4] Sensicast.businesswire.com/releases/sensinet/system/ prweb521310.html

[5] Chenyang Lu, Brian Blum, Tarek Abdelzaher, John Stankovic, and Tian He.:RAP: A Real-Time Communication Architecture for Large-Scale Wireless Sensor Networks. In: Real-Time Technology and Applications Symposium, San Jose, CA, September 2002

[6] Tian He, John Stankovic, Chenyang Lu, Tarek Abdelzaher.: SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks. In: International Conference on Distributed Computing Systems, Providence, Rhode Island, May 2003

[7] Octav Chipara, Zhimin He, Guoliang Xing, Qin Chen, Xiaorui Wang, Chenyang Lu, John Stankovic, _Tarek Abdelzaher.: Real-time Power-Aware Routing in Sensor Networks. In: Fourteenth IEEE International Workshop on Quality of Service(IWQoS 2006)June 19-21, 2006 Yale University, New Haven, CT, USA

[8] Murat Demirbas.: Wireless Sensor Networks for Monitoring of Large Public Buildings. In: First China-US Workshop on Protection of Urban Infrastructure and Public Buildings against Earthquakes and Manmade Disasters

[9] John porter, Peter arzberger, Hans-werner braun, Pablo bryant, Stuart gage, Todd hansen, Paul hanson, Chau-chin lin, Fang-pang lin, Timothy kratz, William michener, Sedra shapiro, and Thomas Williams.: Wireless Sensor Networks for Ecology. In: American Institute of Biological Sciences Volume 55, Number 7, July 2005 pp. 561-572(12)

[10] C.M.Krishna and Kang G. Shin.: Real –Time Systems. In: McGraw-Hill International Editions, 1997

IEEE 802.15.4 Throughput Analysis under IEEE 802.11 Interference

Nazif Cihan Taş
Siemens Corporate Research
Princeton, NJ, 08540
Nazif.Tas@Siemens.com

Chellury (Ram) Sastry
Siemens Corporate Research
Princeton, NJ, 08540
Chellury.Sastry@Siemens.com

Zhen Song
Siemens Corporate Research
Princeton, NJ, 08540
Zhen.Song@Siemens.com

***Abstract*— It is well known that both the IEEE 802.15.4 wireless standard for low power, low data-rate sensor networks, and the IEEE 802.11 wireless local area data networks, operate in the 2.4GHz industrial, scientific and medical (ISM) band. If the networks are in physical vicinity of each other to cause packet interference, most often, it is the 802.15.4 traffic that is adversely affected given the low operational output power of 802.15.4 nodes. In this paper, we undertake a theoretical study on the effect that 802.11 nodes have on the channel utilization capacity of an 802.15.4 network. We suggest what parameters in an 802.11 network can be adjusted dynamically so as to optimize channel/bandwidth utilization of the 802.15.4 network while at the same time minimizing interference. We also present the results of MATLAB simulations of our theoretical framework.**

I. Introduction

The IEEE 802.15.4 wireless standard addresses the medium access and physical layer (MAC/PHY) needs of low data-rate, low-power wireless sensor networks (WSN) or personal area networks (PAN). Its operational frequency includes the 2.4GHz industrial, scientific and medical (ISM) band to facilitate worldwide availability. Given the high costs of deploying and maintaining a wired infrastructure, and in contrast, the cost of IEEE 802.15.4 radio chips rapidly plummeting in recent years, WSN-based applications are becoming increasingly ubiquitous across a range of vertical market segments including building automation, industrial automation, and remote vital-sign monitoring of patients through wearable sensors. At the same time, the ISM band is also occupied by enterprise-level, wireless local area data networks (based on IEEE 802.11b/g standards, also referred to as WiFi), the numbers of which continue to explode. IEEE 802.11[1] is the de facto standard for wireless LANs in the market today with approximately 213 million WiFi chipsets shipped in 2006 [20]. Shown in Figure 1 [10] (is the channel layout of IEEE 802.11 and IEEE 802.15.4 respectively

As can be seen from Figure 1, IEEE 802.11 has 11 channels (1 through 11 between 2.401 GHz and 2.473 GHz each with a bandwidth of 5 MHz and an interchannel spacing of 5 MHz), while IEEE 802.15.4 has 16 channels (11 through 26 between 2.4 GHz and 2.4835 GHz each with a bandwidth of 2 MHz and an interchannel spacing of 5 MHz). Clearly, there is significant overlap between most of the channels offered by the two standards. Furthermore, unlike IEEE 802.15.4 networks,

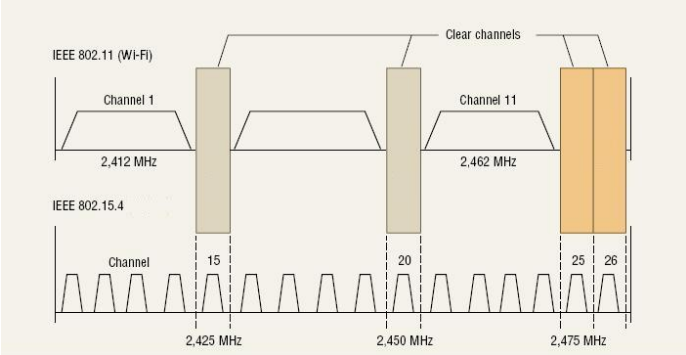


Fig. 1. Channel layout in IEEE 802.11 and IEEE 802.15.4 wireless standards.

WiFi networks involve large data-rates and high power; typical operating power of an IEEE 802.15.4 node is 0 dBm [5], [12], while that of a WiFi node is 30 dBm [1]. As a result, an 802.15.4 node operating in a channel, say channel 1, in the vicinity of an 802.11 node, will effectively be drowned out as a result of the massive disparity in operating powers. Thus, given the possible interference between 802.15.4 and 802.11 networks in close proximity of each other, the issue of managing the co-existence of these networks is crucial in order to maintain the quality of service requirements of the respective applications, particularly IEEE 802.15.4 applications given their low power output.

Most approaches to addressing this co-existence issue involve a channel utilization strategy, i.e., the networks operate on distinct channels that are spaced further apart in frequency and do not overlap. As shown in Figure 1, if the two networks operate on so called "clear channels"; channels 1, 6, and 11 for WiFi networks, and channels 15, 20, 25, 26 for 802.15.4 networks, there will be no interference. While this strategy does indeed result in co-existence, we argue in this paper that this may not be an efficient strategy in terms of available channel and bandwidth utilization, especially as the number of WSN/PAN and WiFi networks increase in the vicinity of each other. What we need is a dynamic adjustment of the IEEE 802.15.4 and IEEE 802.11b/g parameters as a function of interference in order to achieve co-existence and at the same

time optimize channel and bandwidth utilization.

As a first step towards achieving this goal, in this paper, we analytically study the effect WiFi nodes have on the channel utilization capacity of an 802.15.4 network. We then suggest what parameters in an 802.11 network be adjusted dynamically so as to optimize channel/bandwidth utilization of the 802.15.4 network while at the same time minimizing interference. We also present the results of MATLAB simulations of our theoretical framework.

The rest of this document is organized as follows: In Section II, we report on the related works. In Section III, we give background information on IEEE 802.15.4 and IEEE 802.11 internal mechanisms. In Section IV, we illustrate our throughput analysis and in Section V, we present our numerical results simulated. And, finally, in Section VI, we convey possible future work directions.

II. Related Work

802.15.4 has been analyzed before for throughput performance. [16] analyzes the slotted-CSMA version using the Markovian chains as the authors in [3] perform for 802.11. Similarly, [17] focuses on the unslotted-CSMA version. Yet, unslotted-CSMA has been analyzed before in the seminal work [15] which assumed aggregate Poisson arrival of packets. In our work, also, we use this model for our analysis.

Even though the number of works in this area is not significant, WLAN-WPAN coexistence issues has been addressed before. [7], [13] study the WLAN effect on Bluetooth links. [11] determines the best 802.11 channels to use in order to minimize the 802.11 effect on 802.15.4 traffic. WLAN performance under WPAN traffic is analyzed for Bluetooth traffic [8] and for 802.15.4 [9].

802.15.4 performance under 802.11 traffic is analyzed in [18] from packet error rate perspective only. In [19], it is shown experimentally that in case of overlapping channels, more than 92% of WPAN frames are destroyed by WLAN traffic. Similar results are obtained in [6].

III. Background

A. IEEE 802.15.4

IEEE 802.15.4 is the wireless standard developed for ultra-low power, low rate wireless personal area networks (WPANs). IEEE 802.15.4 standard defines the physical and medium access control layer characteristics for WPANs and leaves the other layer details to the users. In addition, as the ZigBee [22] effort that is supported by the industry for standardizing the infrastructure for low-power applications adopted the 802.15.4 standard for their physical and medium access layers, the importance of 802.15.4 standard is escalated. In this section, we give a brief introduction to the 802.15.4 standard, for more information please see [4] and [23].

IEEE 802.15.4 physical layer defines two different options for communication: 2.4 GHz and 868/915 MHz. 868 MHz band is available in Europe whereas 915 MHz is available in the United States; 2.4 GHz is available worldwide. Both of these choices use Direct Sequence Spread Spectrum for modulation. Table I summarizes features of different physical layer choices. Notice that the lower data rate physical choice has better receiver sensitivity because of its lower bit rates.

Data Rate	Code Length	Modulation	Symbol Rate	Bits/ Symbol
1 Mbps	11(BarkerSeq.)	BPSK	1 MSps	1
2 Mbps	11(BarkerSeq.)	QPSK	1 MSps	2
5.5 Mbps	8 CCK	QPSK	1.375 MSps	4
11 Mbps	8 CCK	QPSK	1.375 MSps	8

TABLE II
IEEE 802.11B Data Rate choices.

IEEE 802.15.4 medium access mechanism utilizes the very well known CSMA/CA algorithm very similar to 802.11 MAC layer but without the RTS/CTS support because of the low data rate requirement for WPANs. There are two modes defined by MAC protocol:

- *Beacon-enabled Mode* In this mode, the coordinator periodically transmits beacons in order to synchronize the wireless nodes and identify the PANs. Two subsequent beacons define a beacon superframe during which the wireless nodes can transmit. A superframe is divided into two parts: *Contention Access Period*, during which the sensors nodes content to access the channel, and the *Inactive Period*, during which the transmissions are not allowed and the wireless nodes are expected to stay idle and they can execute sleep states. In this mode, wireless stations use slotted CSMA/CA.
- *Beaconless Mode* In this mode, there is no synchronization among nodes and the devices use the simple unslotted CSMA/CA method.

In the rest of this document, we use the terms 802.15.4 and IEEE 802.15.4 interchangeably.

B. IEEE 802.11

IEEE 802.11 is the standard specification defined for wireless connectivity for fixed, portable, and moving stations within a local area. Similar to its low-power counterpart IEEE 802.15.4 mentioned in the previous section, IEEE 802.11 also defines the PHY and MAC layers for communication. Unlike 802.15.4, however, 802.11 focuses on higher data rate communication as this is more common in local area networks (LANs). Even though there are variations of 802.11 for different purposes, in this document, we focus on the 802.11b as it is the first widely accepted standard. However, our analysis is easily extendible for other variations as well. In the rest of this document, we use the terms 802.11 and IEEE 802.11b interchangeably.

802.11 defines four different data rates with different physical characteristics for diverse environments. Table II summarizes these four different PHY choices. Notice that all PHY choices defined in 802.11 operate on 2.4 GHz band and share this band with other traffic such as 802.15.4 and Bluetooth traffic and with devices such as microwave ovens.

PHY	Band	Data Parameters			Channels	Receiver Sensitivity
		Bit Rate (kb/s)	Symbol Rate (kbaud)	Modulation		
868 MHz	868.0-868.6 MHz	20	20	BPSK	1	$-92dBm$
915 MHz	902.0 - 928.0 MHz	40	40	BPSK	10	$-92dBm$
2.4 GHz	2.4 - 2.4835 GHz	250	62.5	16-ary orthogonal	16	$-85dBm$

TABLE I
IEEE 802.15.4 PHY CHARACTERISTICS.

802.11 MAC layer defines two different access schemas: mandatory contention-based *Distributed Coordination Function* (DCF) for distributed access and optional *Point Coordination Function* (PCF) which provides contention-free access to support applications that require real-time service. As PCF is not widely implemented in the commercial products today, in this document, we focus on the DCF functionality.

DCF operates as follows: any wireless station ready to transmit, first, senses the channel to avoid collisions. If the medium is not idle, the station defers the transmission employing a binary exponential backoff schema. If the medium has been idle for longer than the DIFS (DCF interframe space) amount of time, the transmission can start immediately, otherwise the station picks a random number from the interval $[0, W-1]$ and waits this number of idle slots to appear on the medium where W is the window size and defined as a configuration parameter (IEEE 802.11b default is 32). If transmission attempt is unsuccessful, the window size is doubled until a predefined maximum (IEEE 802.11bs default is 1024) and the process is repeated until the maximum retry count is reached, after which the upper layers are reported about bad channel conditions. In order to avoid channel capture, each successful transmission is followed by a backoff operation.

In 802.11, each transmission packet is acknowledged by the receiver after SIFS (short interframe space) amount of time. In order to give higher priority to acknowledgement packets, SIFS is smaller than DIFS.

IV. THROUGHPUT ANALYSIS

Figure 2 illustrates the deployment schema we are assuming in this document. In this deployment, there is a wireless sensor network using 802.15.4 as its communication schema with constant mean packet arrival rate and there is a WiFi node which transmits 802.11 packets according to a packet arrival rate with constant mean. In order to model this deployment, we considered the distance between the 802.11 node and the center of the sensor network. Indeed, for large number of sensors deployed densely in a region, intuitively, the average interference amount can be calculated using the distance between the interferer and the center of interference region.

A. Interference Analysis

The interference analysis based on extensive field trials of IEEE 802.15.4 network operation in the presence of 802.11 traffic have been reported in literature [14], [10]. In these trials, different 802.15.4 channels were chosen and the resulting performance (measured in terms of packet delivery success rate and percentage of duplicate packets) in the presence of 802.11 nodes were examined.

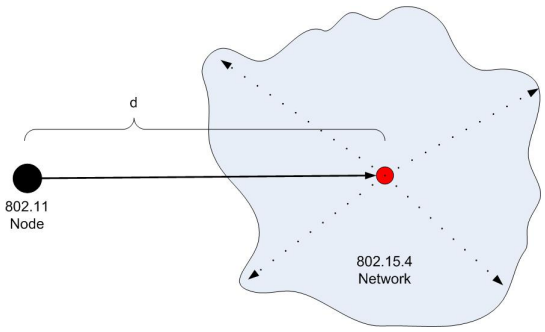


Fig. 2. Heteregenous Networks deployment schema.

The packet delivery rates clearly show that when the 802.15.4 and 802.11 channels overlap, packet delivery rate of 802.15.4 nodes is reduced from 100

Thus, as pointed out in the Introduction, the results in [14], [10] do confirm that the interference 802.11 nodes in close proximity have on an 802.15.4 network is minimal or non-existent when the networks operate on non-overlapping channels, or when the physical distance between two networks is large.

B. Channel Utilization

Let G_{11} denote the arrival rate of packets belonging to 802.11 traffic, including the retransmissions for this class; and similarly let G_{15} denote packet arrival rate of 802.15.4 users. Packet lengths of 802.11 and 802.15.4 are denoted as τ_{11} and τ_{15} respectively.

If we assume that the packet generation is close to exponential distribution, then, the probability that, for instance, an 802.11 packet does not arrive for T amount of time can be approximated as $P_{idle,11}(T) = e^{-G_{11}T}$.

The transmission schema can be analyzed through five different events as shown in Figure 3:

1) T_1 =A successful 802.15.4 transmission.
2) T_2 =A successful 802.11 transmission.
3) T_3 =An 802.15.4 packet collided with an 802.11 packet.
4) T_4 =An 802.15.4 packet collided with another 802.15.4 packet.
5) T_5 =Idle time spent between two 802.15.4 packets.

Then, the channel utilization for 802.15.4 traffic can be represented as

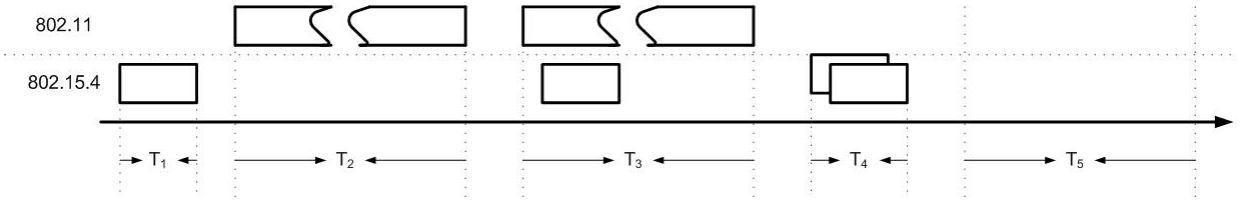


Fig. 3. Possible events that might occur on the medium.

$$S = \frac{T_1}{T_1 + T_2 + T_3 + T_4 + T_5} \tag{1}$$

If a is the propagation delay, then a successful 802.15.4 transmission depends on the fact that, no other 802.15.4 nodes sample the medium for a amount of time and no 802.11 traffic is generated during the transmission. First probability is $e^{-aG_{15}}$ and the second probability is $e^{-G_{11}\tau_{15}}$, then

$$T_1 = \tau_{15} \times G_{15} \times e^{-aG_{15}} \times e^{-G_{11}\tau_{15}} \tag{2}$$

802.11 successful transmission time can be calculated by using the packet generation rate directly. Notice that regardless of the collision probability, if a 802.11 packet is on the medium, the channel medium is wasted from 802.15.4 perspective. Hence, we can merge the second and third events as a 802.11 overall transmission. This value can be directly calculated using the packet 802.11 arrival rate.

$$T_2 + T_3 = G_{11} \times \tau_{11} \tag{3}$$

Fourth event represents the cases where two 802.15.4 packets collide. Notice that a similar analysis for unslotted, nonpersistent CSMA analysis already made in [15] which is based on the assumptions undertaken in [2]. However, as the 802.11 traffic already considered in the second and third evets, we should exclude it here. Hence, the average duration of 802.15.4 collision is

$$\begin{aligned} T_4 \quad = \quad & G_{15} \times (1 - e^{-aG_{15}}) \times e^{-(\tau_{15}+a)G_{11}} \times \\ & \left[\tau_{15} + a + \frac{1}{G_{15}} \times \left(a - \frac{1}{G_{15}}(1 - e^{-aG_{15}})\right)\right] \end{aligned}$$

Finally, the average amount of time that the medium is idle is $\frac{1}{G_{15}}$ in case of 802.15.4 traffic only. If there is 802.11 traffic present at this moment, it counts towards the second event, hence, these cases should be ignored. Then,

$$T_5 = (\frac{1}{G_{15}} - \tau_{15}) \times \alpha \times e^{-G_{11}\alpha(\frac{1}{G_{15}} - \tau_{15})} \tag{4}$$

where α is the normalization factor for changing values of G_{15}.

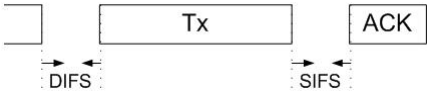


Fig. 4. 802.11 Packet exchange schema.

C. Channel Access Times

A typical 802.11 transmission schema consists of a packet transmission and a corresponding acknowledgement and SIFS amount of time between these two as seen in Figure 4. The user also waits the medium to be idle for DIFS amount of time. However, from our perspective 802.11 traffic occupies the medium during the transmissions only, i.e. for a duration of packet and acknowledgement transmissions. For the sake of simplicity, we will assume that during a 802.11 transmission, channel will look busy even during the SIFS amount of time mentioned above. Hence,

$$\tau_{11} = T_x + T_{ACK} + SIFS \tag{5}$$

ACK size for 802.11b is 38 bytes and SIFS is $10\mu s$. If the packet size for 802.11 traffic is L_{11} bits and the 1 Mb/s data rate is assumed then,

$$\tau_{11} = (L_{11} + 38 + 10) = (L_{11} + 48)\mu s \tag{6}$$

If TCP/IP default value of 1500 bytes is used, $\tau_{11} = 12048\mu s$.

In order to calculate τ_{15}, we should consider only the packet transmission time itself with the assumption that acknowledgement option is turned off. As 802.15.4 data rate is 250 kb/s, if the TinyOS [21] default value of 38 bytes is used, $\tau_{15} = \frac{38*8}{250} = 1216\mu s$.

V. Numerical Analysis

In order to understand the effects of 802.11 traffic on 802.15.4 throughput, we numerically simulated the above functions in MatLab. In this section, we report our findings.

A. Performance Analysis of Heterogeneous Networks

In our first experiment, we report on the effects of 802.11 packet generation rate on the performance. In Figure 5, we see that as the packet generation rate of 802.11 increases, as expected, the channel utilization of 802.15.4 decreases greatly. Notice that. as the packet generation rate of 802.15.4 increases, this diminising effect becomes more and more acute.

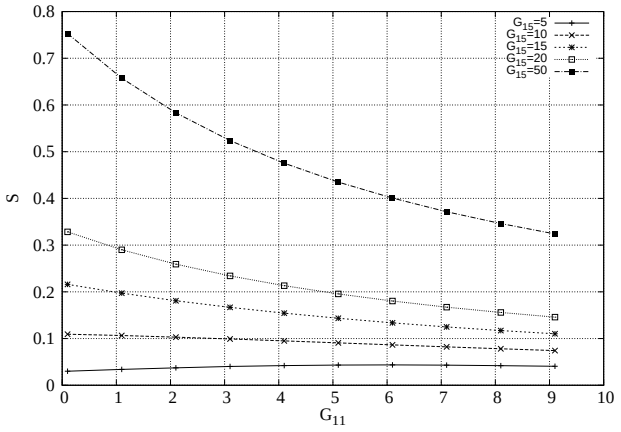


Fig. 5. G_{11} effect on S. $a = 1$.

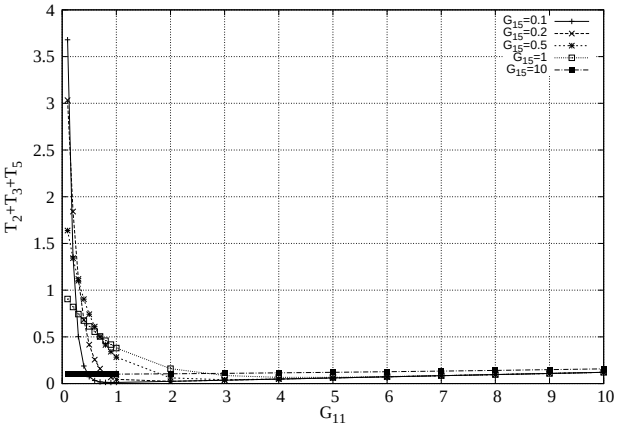


Fig. 6. Cost behavior for changing 802.11 traffic.

One interesting result is that when the packet generation rate of 802.15.4 nodes is too low, increasing 802.11 packet generation rate, surprisingly, increases the channel utilization factor. This anomaly occurs because of the low 802.15.4 traffic rate and can be explained easily by the fact that the channel idle time decreases greatly when the 802.11 traffic increases. Indeed, as can be seen in Figure 6, increasing the 802.11 traffic decreases the sum of 802.11 medium access time and the idle time for low G_{15}. Under normal conditions, this would not affect the 802.15.4 channel utilization as 802.15.4 traffic generation rate is kept constant. However, when traffic rate of 802.15.4 stations is very low, the anomaly explained in Figure 8 happens.

Lets say that the top packet generation schema happens when the 802.11 traffic rate is low where the window size is t and increasing this rate decreases the window size of consideration as shown in Figure 6. In this case, the new window size becomes t' where $t' = t - \Delta$. Notice that since G_{15} is very small $G_{15} \times \Delta \simeq 0$ and the number of packets received in the intervals $[0, t]$ and $[0, t']$ are the same. Hence, the amount of throughput received per second in the latter case

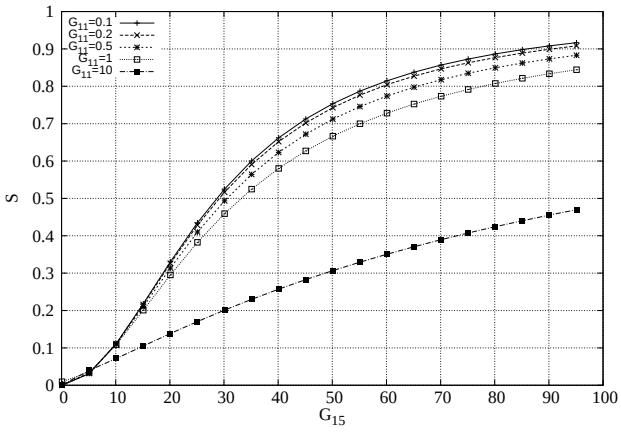


Fig. 7. G_{15} effect on S. $a = 1$.

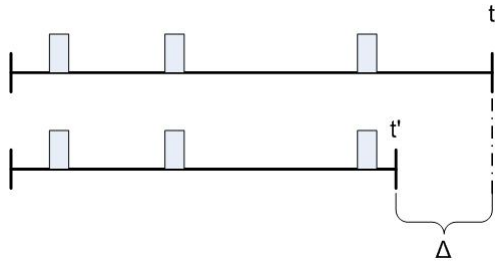


Fig. 8. Channel utilization anomaly.

is $\frac{G_{15} \times t}{t'}$. Then,

$$\begin{aligned} \frac{G_{15} \times t}{t'} &= G_{15} \times \frac{t' + \Delta}{t'} \\ &= G_{15} \times \left(1 + \frac{\Delta}{t'}\right) \end{aligned}$$

which greatly depends on the $\frac{\Delta}{t'}$ factor and as t' gets smaller, this factor starts to get bigger.

In a similar analysis, we looked at 802.15.4 packet generation effect on channel utilization as presented in Figure 7. As the graph suggests, 802.11 traffic rate has indeed a huge effect on channel utilization. As the packet generation rate of 802.11 increases, the channel utilization drop becomes sharper.

As the equations defined above depend on the propagation delay parameter a, we conducted experiments in order to see the effect of a on the channel utilization. In Figure 9, we see this effect for very large propagation delays. Of course, such numbers are not realistic but this figure confirms the intuition that as the propagation delay increases, the channel utilization decreases causing more number of collisions. However, Figure 10 illustrates that for small values of propagation delay (which is at the microseconds level for wireless communication), the channel utilization is independent of the propagation delay.

Notice that the equations defined in the previous sections rely on packet generation patterns as well as channel occupa-

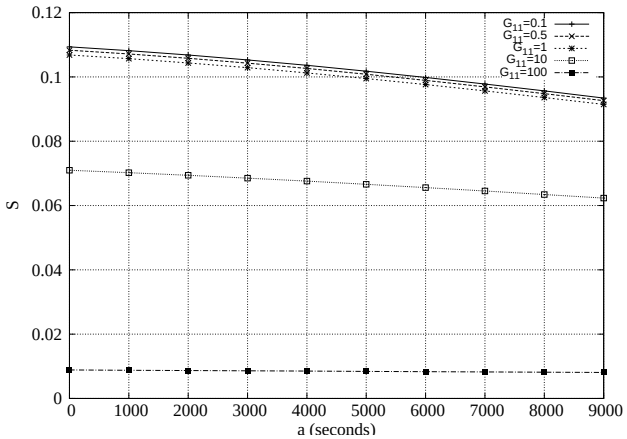


Fig. 9. a effect on S. $G_{15} = 10$.

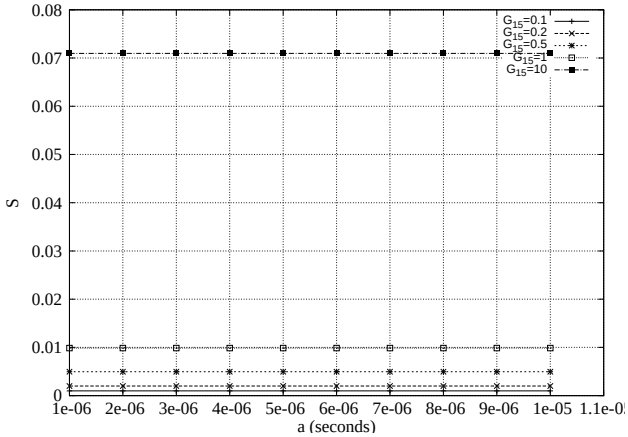


Fig. 10. a effect on S. $G_{11} = 10$.

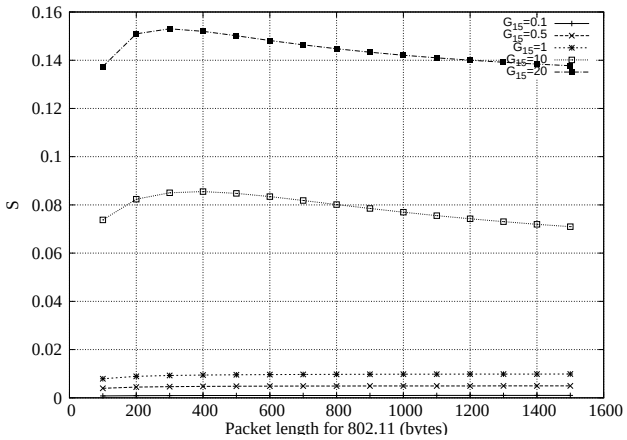


Fig. 11. 802.11 Packet length effect on S. G_{11} normalized according to default value $G_{11} = 10$ for packet length of 1500 bytes.

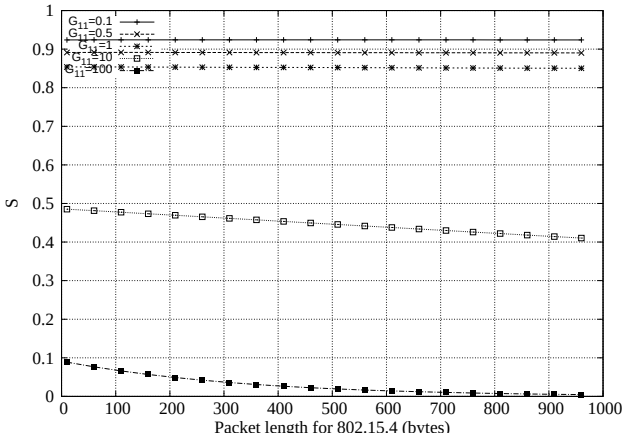


Fig. 12. 802.15.4 Packet length effect on S. G_{15} normalized according to default value $G_{15} = 10$ for packet length of 38 bytes.

tion times. Hence, for a given traffic pattern, we investigated the effect of changing the packet length on the channel utilization. Note that changing the packet length has an indirect effect on the packet generation rate also. For instance, for a given traffic pattern G and channel access time τ, if changing the packet length drops the channel access time to $\tau/2$, this, essentially means that the traffic pattern also increases roughly to $2G$.

Investigation of the 802.11 packet length variation is illustrated in Figure 11. This figure suggests that, initially, as the packet length increases for 802.11 traffic, channel utilization increases since this essentially means to decrease the packet generation rate of 802.11. However, after some point, this increase starts to have detrimental effect on the channel utilization as using larger blocks of accesses yield to larger but more infrequent gaps for 802.15.4 traffic even though the cumulative duration of gaps is the same for both scenarios.

Figure 12 demonstrates a similar analysis for 802.15.4 packet length. Even though this figure suggests using smaller packet lengths in order to maximize channel utilization, since lowering the packet length essentially means to increase the packet generation rate, this modification is not feasible for wireless sensor network which, generally, have very strict duty cycle requirements. Hence, we do not consider packet length adjustment procedures for 802.15.4 in this paper.

IEEE 802.11b standard provides four different modulation techniques to choose from for backward compatibility. Namely, these modulation techniques support 1 Mb/s, 2 Mb/s, 5.5 Mb/s and 11 Mb/s traffic. The decision for the modulation technique is solely left to the user and the standard does not provide a mechanism for this decision. We experimented with different modulation techniques in order to see the level of their effects on 802.15.4 channel utilization. As Figure 13 suggests, using the higher data rate modulations increase the channel utilization as this yields to shorter medium occupancies for 802.11 packets.

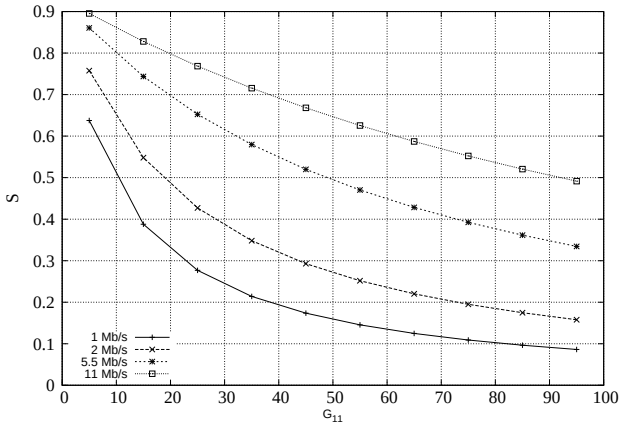


Fig. 13. Physical modulation effect on S. $G_{15} = 100$, $a = 1$, packet lengths are 1500 and 38 bytes for 802.11 and 802.15.4 respectively..

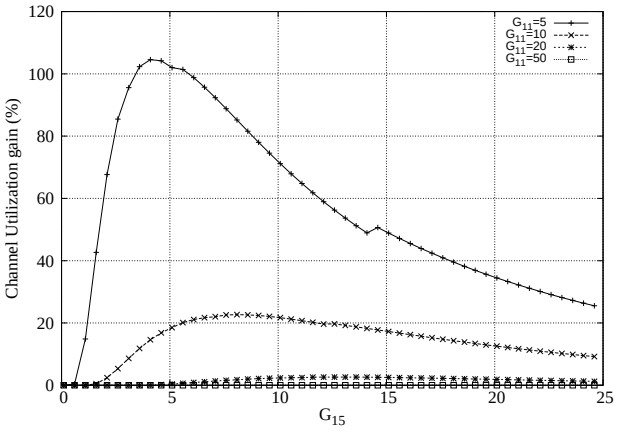


Fig. 15. Channel utilization gain using optimal packet length for 802.11 traffic.Packet length for 802.15.4 is 38 bytes, $a = 1$.

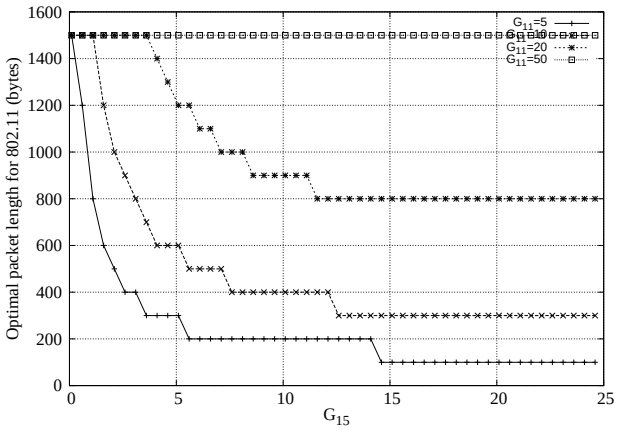


Fig. 14. Optimal packet length to use for 802.11 traffic for different 802.15.4 traffic.Packet length for 802.15.4 is 38 bytes, $a = 1$.

B. Optimal 802.11 Packet Length Calculation

Notice that in Figure 11 the 802.11 packet length change effect on channel utilization is a parabolic shape suggesting there is a point in the function which maximizes the output function, in this case the channel utilization. This shape suggests that there is an optimal 802.11 packet size choice which maximizes the channel utilization for given traffic patterns. In order to see the effect of 802.11 packet size, we further developed another experiment. For this experiment, we looked at all the 802.11 packet size possibilities for a given packet generation pair, and calculated the packet size which maximizes the channel utilization. For instance, in Figure 11 for $G_{15} = 20$ and $G_{11} = 10$, the packet length that maximizes S is around 300. We plotted this function in Figure 14.

Figure 14 suggests that if the packet generation rate of 802.11 is too large than the packet length does not have any effect on the channel utilization whatsoever, which is reasonable since the 802.11 traffic would be very dominant on the medium in such a case. However, for reasonable 802.11 packet generation rates, we see that according to the 802.15.4 traffic, the optimal 802.11 packet size changes. As can be seen from Figure 14, as the 802.15.4 traffic increases, it is always better to use smaller and smaller packets. This, in fact, is meaningful since the channel would be captured by the 802.11 traffic for long times if the 802.11 packet lengths are big. Under these circumstances, it is always better to use smaller packets and increase the packet generation rates for 802.11 traffic.

Figure 15 presents the channel utilization that is gained using the optimal packet length found in Figure 14. The gain calculation is simply

$$Gain = \frac{S_{opt} - S_{norm}}{S_{norm}} \times 100 \tag{7}$$

where S_{opt} is the channel utilization observed using optimal 802.11 packet length whereas S_{norm} is the channel utilization observed using the standard packet length of 1500 bytes.

Notice that for low 802.11 traffic, using optimal packet length has a huge effect on 802.15.4 performance, and the channel utilization is doubled for some 802.15.4 traffic patterns. As the 802.11 traffic rate increases, the medium is captured mostly by this traffic and the outcome of using optimal 802.11 packet length starts to lose its importance. In addition, as the packet generation rate of 802.15.4 increases, the effect of optimal packet length also decreases since, then, it is not enough to partition the 802.11 access on the medium.

VI. Conclusions and Future Work

In this paper we studied one of the most important and challenging wireless issues of our time, namely, the coexistence of 802.11 and 802.15.4 networks beyond a channel separation strategy. This problem is of particular importance given the rapid growth of both 802.15.4-based wireless sensing applications, as well as 802.11-based computer networks; both operating within an enterprise in close proximity of each other. The first goal we set out to achieve is to mathematically characterize the effect that interfering 802.11 nodes have on the

performance (channel/bandwidth utilization) of an 802.15.4 network. Our analysis shows that indeed there are parameters of an 802.11 network that could be dynamically configured so as to optimize the performance of an 802.15.4 network without compromising on the performance of the 802.11 network. The key contribution of this paper was to show that for moderate 802.11 traffic, there is an optimal 802.11 packet length for which the channel utilization of any 802.15.4 network in its proximity doubles when compared with what it would be for normal 802.11 packet length of 1500 bytes. While this result may have some practical utility, our future work involves the extension of the theoretical analysis presented in here so as to study the performance of an 802.15.4 network as a function of several configurable parameters of both 802.15.4 and 802.11 networks. Such an analysis will surely facilitate a robust planning strategy while deploying such networks with large numbers of nodes in the future where just a channel separation plan will clearly not suffice.

REFERENCES

[1] *IEEE 802.11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.*, 1999.

[2] N. Abramson. The ALOHA System–Another Alternative for Computer Communications. *Cluster Computing*, 5:187–201, 1970.

[3] Giuseppe Bianchi. Performance Anaylsis of the IEEE 802.11 Distributed Coordination Function. *IEEE Journal on Selected Areas in Communications*, Vol. 18, Number 3:535–547, March 2000.

[4] Ed Callaway, Paul Gorday, Lance Hester, Jose A. Gutierrez, Marco Naeve, Bob Heile, and Venkat Bahl. Home networking with IEEE 802.15.4: a developing standard for low-rate wireless personal area networks. *Communications Magazine, IEEE*, 40(8):70– 77, Aug 2002.

[5] Chipcon. *CC2420 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver Data Sheet*.

[6] N. Golmie, D. Cypher, and O. Rebala. Performance Evaluation of Low Rate WPANs for Sensors and Medical Applications. *Proceedings of Military Communications Conference (MILCOM 2004)*, 2004.

[7] Jaap C. Haartsen and Stefan Zrbes. Bluetooth voice and data performance in 802.11 DS WLAN environment. Ericsson SIG publication.

[8] I. Howitt. WLAN and WPAN coexistence in UL band. *Vehicular Technology, IEEE Transactions on*, 50:1114–1124, 2001.

[9] I. Howitt and JA Gutierrez. IEEE 802.15.4 low rate-wireless personal area network coexistence issues. *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, 3, 2003.

[10] Terry Hubler. Worry-free wireless networks. `http://www.us.sbt.siemens.com/bau/products/Wireless/HPACEprint.pdf`.

[11] Terry Hubler and Brian Petted. Wireless Networks in Building Automation Systems. Technical report, Siemens Building Technologies, Inc., July 2005.

[12] IEEE Computer Society. *IEEE Standards Department, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, 2003.

[13] IEEE Standards Department,. *IEEE Standards Department, 802.15.2 Part 15.2: Coexistence of Wireless Personal Area Networks with Other Wireless Devices Operating in Unlicensed Frequency Bands*, 2003.

[14] Crossbow Inc. Avoiding RF interference between WiFi and Zigbee. `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/ZigBeeandWiFiInterference.pdf`, 2007.

[15] L. Kleinrock and F. Tobagi. Packet Switching in Radio Channels: Part I–Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics. *Communications, IEEE Transactions on [legacy, pre - 1988]*, 23:1400–1416, 1975.

[16] T.R. Park, T.H. Kim, J.Y. Choi, S. Choi, and W.H. Kwon. Throughput and energy consumption analysis of IEEE 802.15.4 slotted CSMA/CA. *Electronics Letters*, 41:1017–1019, 2005.

[17] Iyappan Ramachandran, Arindam K. Das, and Sumit Roy. Analysis of the contention access period of ieee 802.15.4 mac. *ACM Trans. Sen. Netw.*, 3(1):4, 2007.

[18] S.Y. Shin, S. Choi, H.S. Park, and W.H. Kwon. Packet Error Rate Analysis of IEEE 802.15.4 under IEEE 802.11b Interference. *Proc. WWIC2005*, pages 279–288, 2005.

[19] A. Sikora. Coexistence of IEEE802.15.4 (ZigBee) with IEEE802.11 (WLAN), Bluetooth, and Microwave Ovens in 2.4 GHz ISM-Band. Technical report, Steinbeis-Transfercentre, 2004.

[20] Gemma Tedesco. The Wi-Fi Chipset Market: Portable Connectivity Applications Drive Volumes. In-Stat Market Research Report, May 2007.

[21] TinyOS Website. http://www.tinyos.net/.

[22] Andrew Wheeler. Commercial Applications of Wireless Sensor Networks Using ZigBee. *Communications Magazine, IEEE*, 45(4):70–77, 2007.

[23] Jianliang Zheng and Myung J. Lee. Will IEEE 802.15.4 make ubiquitous networking a reality?: a discussion on a potential low power, low bit rate standard. *Communications Magazine, IEEE*, 42(6):140– 146, June 2004.

Efficient Routing for Vehicular Networks based on Relative Position and Velocity

Qing Yang, Raghu Kisore Neelisetti, Alvin Lim, *Member, IEEE*, Prathima Agrawal, *Fellow, IEEE*

***Abstract*—Efficient and practical communications between large number of vehicles are critical in providing high level of safety and convenience to drivers. Crucial real-time information on road hazard, traffic conditions and driver services must be communicated to vehicles rapidly even in adverse environments, such as "urban canyons" and tunnels. We propose a novel routing protocol in vehicular networks that does not require position information (e.g. from GPS) but instead rely on relative position that can be determined dynamically. This *GPS-Free* Geographic Routing (GPSFR) protocol uses the estimated relative position of vehicles and greedily chooses the best next hop neighbor based on a *Balance Advance (BADV)* metric which balances between proximity and link stability in order to improve routing performance. In this paper, we focus primarily on the complexity of routing in highways and solve routing problems that arise when vehicles are near interchanges, curves, and merge or exit lanes of highways. Our simulation results show that by taking relative velocity into account, GPSFR reduces link breakage to only 27% that of GPSR in the dense network. Consequently, GPSFR outperforms GPSR in terms of higher data delivery ratio, lower delay, less sensitivity of the network density and route paths' length.**

***Index Terms*—Relative positions, Routing, GPS free, Vehicular ad hoc networks, Inter-vehicle communication**

I. Introduction

IN the future, large scale vehicular ad-hoc networks will be available to provide drivers with higher level of safety and convenience. For instance, multi-hop wireless communication between vehicles can enhance ACC (adaptive cruise control) systems by enabling rapid adaptation of longitudinal control in response to traffic accidents that just occur a short distance ahead of it (possibly a few wireless hops). It can also enable smart vehicles to react rapidly to braking maneuvers when a "hidden" vehicle that is ahead of it by several vehicles is braking. Such capabilities will be instrumental in improving highway traffic safety. As reported in [12] by the National Highway Traffic Safety Administration (NHTSA), in U.S. alone, vehicle crashes on the highway resulted in the loss of as many as 40,000 lives and an overall economic losses of more than $230 billion. The motivation to reduce accidents on highways has sparked increasing interest in research on improving vehicle safety through inter-vehicle communication using vehicular ad-hoc networks [2, 3, 6], including an effort by IEEE on a standard for inter-vehicle communication.

There are three possible network architectures of vehicular networks: infrastructure-based, ad-hoc networks and hybrid. In this paper, we focus on vehicular ad-hoc network (VANET) architectures because the cost of building such a network is very low and it can even operate in the events of disasters. Deployment of such networks is flexible and self-organizing. The other architectures require infrastructure support which has three drawbacks: high operating cost, limited bandwidth and symmetric channel allocation for uplink and downlink. There have been a number of research efforts on vehicular ad-hoc networks. For instance, the medium access control (MAC) problem was addressed in [4, 5]. To improve safety and commercial services, a multi-channel MAC protocol was proposed in [6]. Routing issues were addressed in [1, 2, 3, 7], including vehicle-assisted trajectory-based routing protocol [1], mobility-centric data dissemination [2] and position-base routing [7]. To further solve the local disconnection problem, information on vehicle headings can be used to predict a possible link breakage prior to its occurrence and then avoid routing to an imminent disconnected next hop [3].

However, all the above routing protocols rely on the positions of nodes and require vehicles to be equipped with GPS receivers. Though GPS will become standard equipment in vehicles in the future, it may still fail to work if the signals from satellites are affected by tall buildings in "urban canyons", tunnels, in bad weather, or when the power is depleted. In this paper, we present a GPS-free geographic routing (GPSFR) protocol that uses only relative positions of vehicles which can be determined dynamically. Based on the relative distance and velocity, a new routing metric called balance advance (*BADV*) is designed to balance between proximity and link stability. Unlike other route optimization metrics [2, 7], *BADV* improves performance in routing without relying on nodes' locations. Our simulation results of vehicular networks in highway scenarios show that GPSFR outperforms GPSR [16], achieving fewer link breakage, higher data delivery ratio and low network delay.

The remainder of this paper is organized as following. In Section II, we summarize several related work. Then, we discuss our motivation and assumptions in Section III. The

Q. Yang is with the Computer Science and Software Engineering Department, Auburn University, AL 36849 USA (phone: 334-844-6324; e-mail: yangqin@auburn.edu).

R. Neelisetti is with the Computer Science and Software Engineering Department, Auburn University, AL 36849 USA (phone: 334-844-6324; e-mail: neelira@auburn.edu).

A. Lim is with the Computer Science and Software Engineering Department, Auburn University, AL 36849 USA (email: limalvi@auburn.edu).

P. Agrawal is now with the Department of Electrical and Computer Engineering, Auburn University, AL 36849 USA (email: pagrawal@eng.auburn.edu).

Relative Position Maintenance (*RPM*) and routing algorithms are described in Section IV. In Section V, we discuss our simulation environment and present the simulation results. Section VI presents the conclusion and future work.

II. RELATED WORK

There are a number of existing techniques for finding the location of nodes in wireless ad hoc network, such as Receiving Signal Strength Indicator (RSSI), Angle of Arrival (AOA), Time of Arrival (TOA) and Time Difference of Arrival (TDOA). All of these techniques require nodes to be able to measure the distances between itself and the neighbors using signal strength or time differences. Therefore, the effectiveness of this sort of approaches will rely heavily on the accuracy of distance estimation which may be adversely effected by large spurious variation in signal strength and time synchronization, the absence of line of sight, and specialized signal processing hardware or software. In addition, the above techniques require anchor nodes whose positions are obtained from GPS.

A fully distributed, infrastructure free positioning algorithm for mobile ad hoc network has been proposed in [8] that do not rely on the anchor nodes. However, it is unsuitable for vehicular networks for two reasons. First, after determining the relative positions of neighbors, each node must change its local coordinate system to the network coordinate system. Such update overhead increases as the network size increases. In fact, it is proven in [9] that the volume of message exchanged in [8] increases exponentially with the node density. Secondly, in highly mobile vehicular networks, the overhead of updating location reference group composed by nodes with lower moving speed is significant. Although cluster-based method in [9] can generate less communication overheads compared to [8], the number of message exchange is still huge because it requires coordinate translation of master nodes throughout the network.

A scheme was proposed in [10] which can localize nodes through fewer message exchanges. However, the scheme in [10] is applicable to ad-hoc networks with less mobility, such as sensor networks, but unsuitable for vehicular networks. The high mobility in vehicular networks will result in large network overhead because of the periodic bootstrapping beacons. The number of flooding nodes during the bootstrapping phase will increase as network size increases [10]. Relative position is also used to warn if a collision is happening by checking the relative distance between vehicles [11], but has not been used for solving the routing problem.

Unlike all existing methods, the relative position information of nodes in GPSFR can be maintained through only localized broadcasting and hence significantly reduces position update overhead compared with those that require flooding the entire network. In addition, to the best of our knowledge, GPSFR is the first method to improve the performances of geographic routing in VANET without reliance on nodes' position.

III. ASSUMPTIONS

We focus primarily on vehicular networks in the rural highway scenario. A rural highway provides a link between urban areas. To determine and maintain neighbors' relative position, each node requires a compass and two directional antennas [13] pointing in opposite directions. One antenna is for sending/receiving data to neighbors in front of it, and the other for those behind it. Other advantages of directional antennas are longer radio ranges, reduction in exposed stations problems and reduction of co-channel interference. We also assume that the coverage area of each antenna is a semi-circle, thus the area covered by the two antennas will form a circle.

IV. ROUTING PROTOCOL

We now describe the GPS-Free Geographic Routing (GPSFR) protocol. This protocol consists of two parts: *relative position maintenance*, which computes and maintains the relative positions of neighbors without GPS support and *routing algorithm*, which greedily transmits packets based on the novel link metric called *balanced advance (BADV)*.

A. Relative Position Maintenance

While moving in the same direction on highways, vehicles will construct a linear network, as shown in Fig. 1. Problems of vehicles at interchanges and ramps will be discussed later. For now, we will just focus on linear networks. In such networks, the delivery of packets can be categorized as *forwarding* or *backwarding*. In forwarding (backwarding), the routing algorithm needs only to choose the next hop from neighbors that are in the same (opposite) direction as packets being delivered. In order to achieve this, each node has to compute and maintain the relative positions of all its neighbors. As shown in Fig. 1 (a), suppose all nodes from 1 to 6 are neighbors, then from node 3's perspective, node 5 should have a closer relative position than node 6. Also from node 3's perspective, the relative position of node 1 should be further than node 2's.

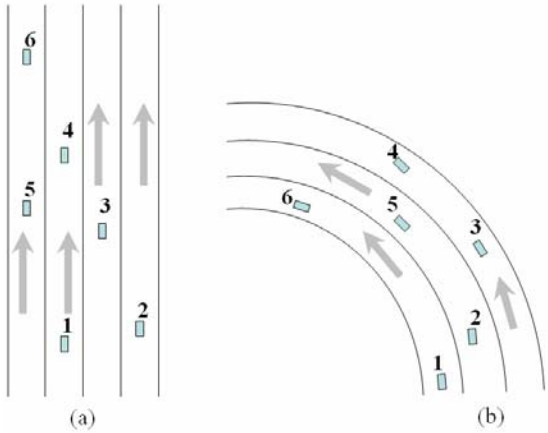


Fig. 1. Illustration of nodes' relative positions on straight and curve road.

Suppose the forward and backward directional antennas are *f_antenna* and *b_antenna*, respectively. If the message arrived at one's *f_antenna*, then it must be sent by a node in front of the receiver; otherwise, it comes from a rear node. Therefore, each

node can divide its neighbors into two groups (*fgroup* and *bgroup*) by checking from which antenna the messages are received. For example in Fig. 1, the *fgroup* of node 3 should be {4, 5, 6} and the *bgroup* is {1, 2}. Even on a curve, as shown in Fig. 1 (b), vehicles can also divide neighbors into two groups (assume the curve is not very sharp which is reasonable on the highway). In addition, we can easily distinguish packets received from the vehicles moving in the opposite direction because if such message was sent from *f_antenna* (or *b_antenna*) of nodes moving at the opposite direction, then the receiver will obtain it also from the *f_antenna* (or *b_antenna*).

After dividing neighbors into proper groups, each node will send such group information periodically. The format of this beacon message (called *group_update*) is: <*bgroup*, *id*, *velocity*, *direction*, *fgroup*>. If no *group_update* was received after a certain time-out period *T*, the neighbor will be considered as out-of-range and deleted from the neighbor list. Actually, we could make GPSFR's beacon mechanism fully reactive, in which nodes will solicit beacons only when they have data to transmit. However, we have not felt it necessary to take this step since the one-hop beacon overhead does not cause too much congestion.

Although computing the relative positions of nodes are straightforward, there is still a *hidden neighbor's* problem. Suppose vehicle A has two neighbors B and C in front/behind, but B and C are not neighbors; then we can say B and C are the *hidden neighbors* of A. For example, as shown in Fig. 2, node 2 and 4 are the *hidden neighbors* of node 1.

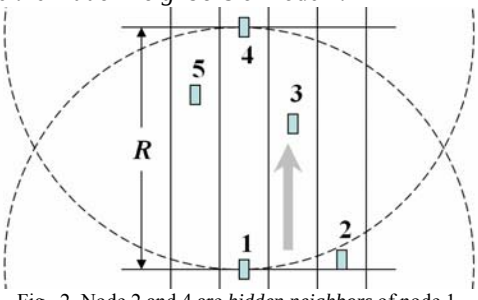


Fig. 2. Node 2 and 4 are *hidden neighbors* of node 1.

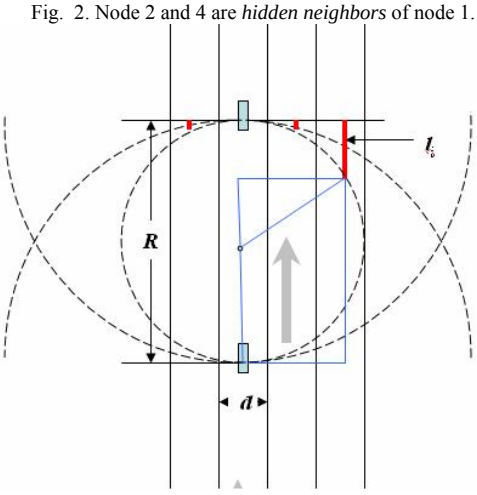


Fig. 3. There are only small pieces where *hidden neighbors* may exist.

Since the *hidden neighbors* are actually not neighbors, then it is not always possible to obtain their relative positions through directional antennas. As shown in Fig. 3, let the communication range of nodes be *R* and the width of each lane be *d*. We can find that on each lane, there is only a small area where *hidden neighbors* may exist. We denote the length of such area as l_s, which can be calculated as follows:

TABLE I
RELATIVE POSITION MAINTENANCE ALGORITHM

Algorithm: Relative Position Maintenance (***RPM***)
Input: Message m_j <GF_j, ID_j, V_j, D_j, GB_j> received from n_j
Output: Ordered link list F_i and B_i for current node n_i
C: Cache for all recently received messages
F_i: Ordered link list of neighbors located in front of n_i
B_i: Ordered link list of neighbors located behind n_i
e: Temp variable holding the element of the ordered link list
GF_i: Neighbors located in front of n_i
GB_i: Neighbors located behind n_i
f-antenna: whether message received from *f* antenna
Interval: Period of beaconing message

1. **if**(m_j is not in *C*) **then**
2. **if**(size of m_j == 2&&D_i==D_j)
3. **if**(*f-antenna*) **then**
4. add ID_j into GF_i;
5. **else**
6. add ID_j into BF_i;
7. **endif**
8. exit
9. **endif**
10. **if** (size of m_j >=3 && D_j is on clock-wise direction of D_i)
11. drop this msg.
12. **endif**
13. add/update the entry of<GF_j, ID_j, V_j, D_j, GB_j> in *C*

/* n_j overtake n_i */

14. **if**(ID_j is in B_i && *f-antenna*) **then**
15. *e*←remove element corresponding to ID_j from B_i
16. set *e* as an anchor and reset *e*'s life time
17. e.position←0;
18. **endif**

/* n_j move backwards of n_i */

19. **if**(ID_j is in F_i && !*f-antenna*) **then**
20. *e*←remove element corresponding to ID_j from F_i
21. set *e* as an anchor and reset *e*'s life time
22. e.position←0;
23. **endif**
24. **if**(*f-antenna*) **then** /* add n_j into the corresponding list */
25. *Insert*(F_i, GF_j, GB_j, ID_j, V_j)
26. **else**
27. *Insert*(B_i, GF_j, GB_j, ID_j, V_j)
28. **endif**
29. update GF_i and GB_i by new F_i and B_i

/* received messages from all neighbors */

30. **if**(one *Interval* passed) **then**
31. *RPU*(V_i, F_i, *true*) and *RPU*(V_i, B_i, *false*)
32. **endif**
33. **else**
34. reset the life time of element related to ID_j in F_i or B_i
35. **endif**

$$l_i = R/2 - \sqrt{(R/2)^2 - i^2 \times d^2} \qquad (i = 1, 2, \cdots, m-1) \quad (1)$$

where m is the number of lanes which is usually from 2 to 6; R is 250 meters and d is 3.6 meters, which is the typical width of lanes in highways [14]. Then the length of each piece will be very short, so the *hidden neighbor* problem is an unlikely event in vehicular networks. Note that the *hidden neighbor* problem also arises when vehicles are side by side, close to an interchange or trying to leave the highway. In our protocol, the *hidden neighbors* are not included in the neighbor list, since they are only hidden for a short time.

TABLE II
INSERT A NODE INTO THE ORDERED LINK LIST AT THE PROPER POSITION

Procedure: *Insert*(L_i, GF_j, GB_j, ID_j, V_j)
L_i: Ordered link list of node n_i
GF_j: Group information of node n_j's front neighbors
GB_j: Group information of node n_j's behind neighbors
ID_j: ID of node n_j
V_j: Velocity of node n_j
added: bool value indicating whether n_j is added successfully
1. *added*←*false*
2. **for** all element *e* in L_i **do**
/* n_j and node *e.ID* are *hidden neighbors* */
3. **if**(*e* is not in GF_j or GB_j) **then**
4. exit
5. **endif**
/* n_j and node *e.ID* are not *hidden neighbors* */
6. **if((***e* in GF_j && L_j is F_i) || (*e* in GB_j && L_j is B_i)) **then**
7. insert < ID_j, V_j> into L_i at the position just before *e*
8. *added*←*true*
9. **endif**
/* L_i is empty or n_j is the farthest one */
10. **if**(*added*!=*true*) **then**
11. append < ID_j, V_j> in L_i
12. **endif**
13. reset the lifetime of < ID_j, V_j> in L_i

1) Relative Position Maintenance (RPM) Algorithm

Within a certain neighborhood, except for those *hidden neighbors*, each node can determine whether or not a neighbor is in front of it through the directional antennas. The Relative Position Maintenance (*RPM*) algorithm uses *group_update* message exchanges to compute and maintain nodes' relative positions. As shown in the TABLE I, each node n_i will maintain two linked lists F_i and B_i. F_i is used to record the front neighbors and B_i is for rear neighbors. Elements of each linked list are ordered by the nodes' relative positions.

If a message is received from n_j, then node n_i will first check whether or not this message is in its cache. If this message matches an entry, then there will be no change on n_j in the list. It just updates the lifetime of node n_j in the list. If it is a new message, n_i will first add/update the message in its cache and then arrange n_j's new relative position in the list. Line 2-9 is used to deal with the scenario of new vehicles merging into the networks, which will be examined later. Line 14-28 arrange node n_j in the corresponding list. We notice that, if node n_j overtook n_i during the last beaconing period, then the current distance between those two nodes may be approximately equal to 0. From n_i's perspective, n_j will be an anchor node since its exact relative distance is now known. In the later beacon periods, we can estimate n_j's new relative distance from the length of beaconing period and the velocity gap between n_j and n_i. Clearly, the relative distances of anchor nodes are more accurate. To calculate others' relative distances, we propose the relative position update (*RPU*) procedure which will be discussed later.

So far, we have established the relative positions of all neighbors, but we do not know the relative distance between them. As described above, some neighbors may become anchor nodes that have more accurate position. Thus, we can use those anchor nodes' relative distance to estimate other distances. If there is no anchor node in the list, then nodes' distances will be estimated to be evenly distributed between each neighbor.

2) Vehicles at Interchanges

On rural highways, there will be many interchanges which are road junctions that typically utilize grade separations, and one or more ramps, to permit traffic on at least one road to pass over or under the highway. Vehicles near an interchange do not need to stop. However, some vehicles may slow down if they are leaving the highway, which will be discussed in the following section. Here, we only focus on vehicles at interchanges that are not exiting. Let a node, n_i, moves through an overpass over a road $road_1$. At the interchange, it may receive packets from nodes moving on the over-passed $road_2$. However, since the directions of those nodes will be different from that of node n_i, these packets are dropped. To safely drop packets, a precondition is that all vehicles are moving on the original road, i.e. there is no steering information in the last few seconds. To implement this, the packet header will contain a flag field that is set if the vehicle steers in the last few seconds.

3) Vehicles Joining Highways

Vehicles can join the highways in only two ways: entrance paths or ramps of the interchange. Regardless of the type, entrances must be located besides the highway at a certain angle. That means vehicles on the entrance will have a different moving direction compared with those already on the highway.

Let us assume node n_j is trying to join the network, and N denotes the set of nodes that were already in the network. To join the network, n_j periodically sends a *join-in* message of the form <*id*, *direction*> until it receives a *group_update* message from responding nodes in the existing networks. After receiving the *join-in* message, each node n_i will first check whether it is moving at the same direction. If so, they add n_j into the corresponding group (*fgroup* or *bgroup*) and then sent out the new *group_update* message to its neighbors; otherwise, it drops this packet. So when fully merged into the highway, n_j can receive all *group_update* messages from nodes in N. Hence by running the *RPM* algorithm, node n_j determines the relative positions and the group information of its new neighbors. Then after one beaconing period, all nodes in N will be able to arrange n_j into the correct link list.

In conclusion, vehicles can participate in the network if and only if they are already in the network. Thus, vehicles on the entrance path or ramp are not considered to be in the network and so packets will not be forwarded out of the highway.

4) Vehicles leaving Highways

If packets need to be forwarded to the front and the forwarder (vehicle) is trying to leave the highway, then this packet will go out of networks. To avoid this problem, a backup scheme is adopted to send the packet to another rear node. The following are the details of this scheme.

While vehicles make a right turn, two cases may occur: this vehicle is on a right-turn curve of the highway, or it is leaving the highway. In both cases, packets are both forwarded and sent to a rear neighbor. If the rear neighbor is also leaving the highway, this backup process continues for a certain threshold number of times. Though this backup scheme can avoid routing packet out of the network, it still has some overhead while the vehicles are moving on a curve. Suppose the backup process is repeated k times, then the problem is how to choose a minimal k.

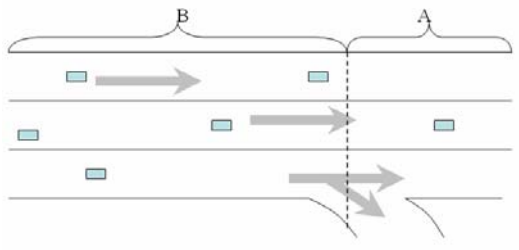


Fig. 4. Vehicles moving around an exit of the highway.

Suppose near the exit of a highway, as shown in Fig. 4, there are m cars connected through wireless links and one of them, e.g. q, is located at the junction between the highway and exit. Then there will be $m + 1$ possible deployment of these nodes. For example, one is in the area A and m - 1 in the area B. If there are one or more nodes in the area A, then the packet will pass the exit successfully. The problem arises only if there is no vehicle in the area A, so the probability of this case will be $\frac{1}{(m+1)}$. Now suppose all m nodes are in area B, and then the probability of k continuous nodes leaving off the highway will be $\frac{1}{C_m^k}$. Therefore, we can obtain that after k times of backup, the probability of packet being routed out of the network is:

$$P_{out} = \frac{1}{m+1} \cdot \frac{1}{C_m^k} \cdot p^k \tag{2}$$

where p is the probability that the node leaves the highways — the largest value for this is 0.5. Now assume we are trying to forward a packet through a routing path with n exit junctions. Then the probability p of the last exit is 0.5, while the probability of the vehicle leaving at the first exit is $\frac{1}{(n+1)}$. Therefore, we can calculate the probability of the packet successfully reaching the destination as:

$$P_{suc} = \prod_{i=1}^{i=n} \left(1 - \frac{1}{m_i + 1} \cdot \frac{1}{C_{m_i}^k} \cdot \left(\frac{1}{i+1} \right)^k \right) \tag{3}$$

It should be noted that sometimes m_i might be smaller than k. In this case, the packet will be routed out of the range. In fact, this failure is caused by a network partition, so it will not be considered in the performance of routing protocol. In most cases, m_i will be bigger than k, so the minimal value of P_{suc} can be obtained when m is equal to k. Now we have

$$P_{suc} > \prod_{i=1}^{i=n} \left(1 - \frac{1}{k+1} \cdot \left(\frac{1}{i+1} \right)^k \right) \tag{4}$$

Let k equals to 3, then the P_{suc} will be within [0.95, 0.96]. In fact, the value of P_{suc} will increase when k increased. However, we believe the value of 0.95 was already large enough for our network routing protocol. Therefore, as shown in TABLE IV of the implementation, we choose $k = 3$.

B. Routing Algorithm

1) Distance Advance

In GPSR [16], the current node n_i greedily selects one neighbor that is closest to the destination as the next hop. The implicit goal of such strategy is to maximize the distance advance and eventually minimize the total number of hops. Let us denote such *distance advance* (*ADV*) of a neighbor n_j as

$$ADV_j = \begin{cases} \frac{d_{ij}}{r_i} & (if \;\; d_{ij} \le r_i) \\ 0 & (if \;\; d_{ij} > r_i) \end{cases} \tag{5}$$

d_{ij} is the relative distance between node n_i and n_j. Whether n_j is behind or in front of n_i, the value d_{ij} is always larger than or equal to zero. Clearly, the conventional geographic routing protocols try to maximize *ADV* of next hop.

2) Balanced Advance (BADV)

Balanced Advance (BADV) aims to avoid choosing an unstable node as the next hop while gaining as much distance advance as possible. The goal of *BADV* is to balance between large distance advance and good link stability.

$$BADV_j = \alpha \cdot ADV_j + (1-\alpha) \cdot e^{-\frac{\Delta v_{ij}}{d}} \; (if \; \Delta v_{ij} < 0, e^{-\frac{\Delta v_{ij}}{d}} = 1) \tag{6}$$

$\Delta v_{ij} = v_j - v_i$ is the velocity difference between n_j and n_i , and d is the distance from n_j's current position to the edge of n_i's communication range. Therefore, suppose t_j is the time used by n_j to move out of n_i's range; then a longer t_j implies a more stable link between n_i and n_j. If Δv_{ij} is less than zero, node n_j is moving closer towards n_i. In this case, we consider the link stability as 1 because such link will increasingly become stronger until n_j move into the different neighbor group of n_i. Since the beaconing period is 1 – 2 seconds, Δv_{ij} within such a short time will not change much. Thus, we can trust this value for at least one beaconing period.

Although the concept of *BADV* is simple, it has many benefits in wireless vehicular networks. First, the data delivery ratio will be increased because of reliable transmission links. If

only distance advance is used, as in GPSR, the link to the selected next hop may suffer from a poor quality due to increased distance. Second, the hit rate of finding next hop's MAC address from cache table will be increased, so times of ARP request and reply will be decreased. Consequently, network delay can be reduced because of fewer retransmissions due to stable links. Third, fewer changes in the next hop will reduce channel switching overhead. For example, if [15] was adopted as the MAC protocol, then there will be a huge time slot allocation overhead due to the frequent next hop change. However, if the data is an emergency message, then GPSFR will use the maximal *ADV* policy by setting alpha as 1, because there is more benefit to choose the shortest path than a stable one. The detailed routing algorithm is shown below in TABLE IV.

TABLE IV
ILLUSTRATION OF THE LOOK-AHEAD ROUTING

Algorithm: Look-Ahead Routing
Input: Current node ID *i* and packet deliver direction *dir*
Output: Next hop's ID *j*
F_i: Ordered link list of neighbors located in front of n_i
B_i: Ordered link list of neighbors located behind n_i
e: Temp variable holding the element of the ordered link list

```
1. if(dir == forwarding) then
2.    if(F_i == NULL) then
3.       carry and forward the msg.
4.       if (turning right during waiting) then
5.          if (counter of the message < 3) then
6.             e←the first element of B_i
7.             if ( e != NULL) then
8.                increase the counter and backup msg. to e
9.             else
10.               carry and forward the msg.
11.            endif
12.         else
13.            drop the msg.
14.         endif
15.      endif
16.   endif
17.   if(F_i != NULL) then
18.      e←node with the largest BADV
19.      forward message to e
20.   endif
21. endif
22. if(dir == back warding) then
23.   if(B_i != NULL) then
24.      e←node with the largest BADV
25.      forward message to e
26.   else
27.      carry and forward the msg.
28.   endif
29. endif
```

V. SIMULATION RESULTS AND EVALUATION

To measure how well we meet our design goals, we use ns2 (ns2.29) to simulate and measure the performance of GPSFR. To compare the performance of GPSFR with the prior work for vehicle communication, we choose the well-known GPSR [16] protocol. Since modeling of complex vehicle movement is important for accurately evaluating protocols, we generated the movement of nodes using VanetMobiSim [17] whose mobility patterns have been validated against TSIS-CORSIM, a well known and validated traffic generator. We focus mainly on the two-lane two-direction highway scenario with different node densities and velocities. Details of the simulation's setup are list in TABLE V.

TABLE V
SIMULATION PARAMETERS

Parameter	Value
Number of nodes	100
Communication range	250m
Velocity	65-80 miles per hour
Packet size	1024 Bytes
Data sending rate	1-8 packets per second
Beacon interval	5.0 seconds
Alpha	0.7

The simulation time is 2000 seconds and each scenario is repeated 20 times to achieve a high confidence level. At each run, arbitrary vehicle pairs were selected as the source and destination. To evaluate the performance on different data transmission density, we vary the data sending rate from 1 to 8 packets per second. The performances metrics are link stability, data delivery ratio and data delivery delay.

A. Link Stability

We measure the number of times of next hop link breakage in GPSFR and GPSR, as shown in Fig. 5. GPSFR always generate less link breakage than GPSR. Network density is defined as the average number of neighbors for every node. In dense networks, the number of link breakages in GPSFR is only 27% of that in GPSR. This is because the next hop selected by maximizing *BADV* will be more stable and fewer next hop changes occur. However, in the sparse network, GPSFR outperforms GPSR to a lesser degree. This is because in spare networks, the number of candidate nodes that can be chosen as the next hop is limited. So GPSFR may have no choice but to choose the same ones as GPSR does. However, it still suffers from fewer link breakages. In Fig.5, the percentage value denotes the probability of velocity change at every vehicle, which is used to model how dynamic the network is. As we can see, the frequent velocity change of vehicles does not affect the link stability of GPSFR too much.

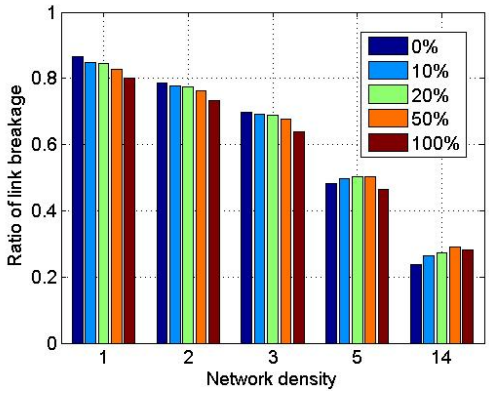


Fig. 5 Percentage of next hop changes various different network densities.

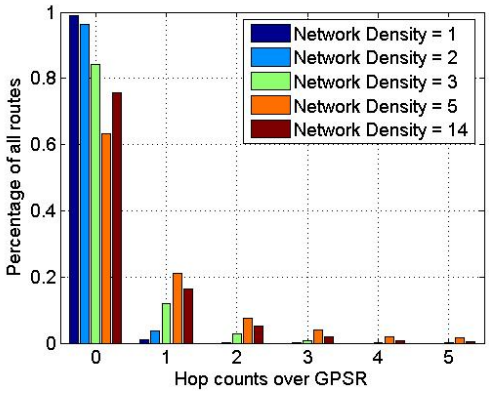


Fig. 7. Path length beyond GPSR when velocity changes at possibility 100%.

B. *Routing Path Length*

Since *BADV* considers the trade-off between stability and distance advance, the length of routing path in GPSFR may be increased because of the slight reduction in distance advance at each hop. Fig. 6 and 7 give the histogram of the extra routing hops of GPSFR compared to that of GPSR. No matter how dense the network is, most of the routes in GPSFR are of the same length as GPSR. In addition, the longer routes are mostly one or two hops more than that of GPSR. Therefore, to maximize *BADV*, we indeed increase the number of hops by just enough to ensure higher data delivery ratio and lower network delay. Fig.6 shows the scenario where all vehicles are in the cruise control (no velocity change). In Fig.7, vehicles change their velocity all the time during the simulation. Note that no matter how dynamic the network, GPSFR always delivers large majority of packets along the path with the fewest number of hops.

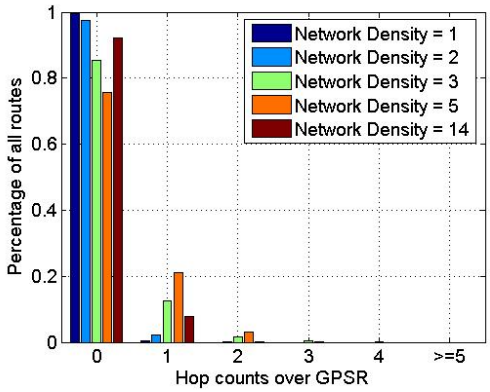


Fig. 6. Path length beyond GPSR when there is no velocity change.

C. *Data delivery ratio*

GPSFR3/GPSR3 denotes that the neighbor time-out period is 3 times the beacon period, while GPSFR/GPSR means that the time-out period is equal to the beacon period. Geographic routing in VANET may suffer from the problem of out-of-date neighbors due to the high mobility of vehicles. One possible solution is to shorten the time-out period of neighbors. Since the neighbors' information is more accurate, higher delivery ratio can be achieved, as shown Fig.8. At each hop, GPSR always try to maximize the distance advance. However, the chosen one may be out of range after a short time. This may increase packet loss. On the other hand, by using *BADV*, GPSFR can predict neighbors' positions before selecting the next hop. So only those that will still be in range after transmission will be considered as candidates. Therefore, the data delivery ratio of GPSFR is higher than that of GPSR.

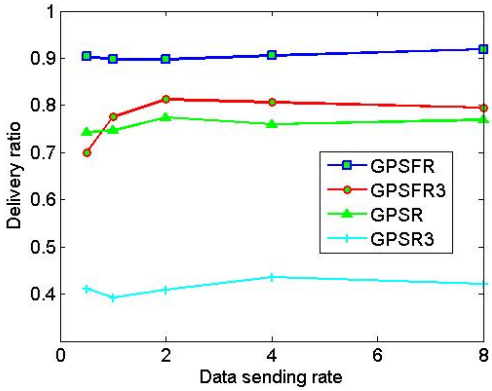


Fig. 8. Data delivery ratio various different data sending rate.

D. *Network delay*

Fig.9 shows that the delay of GPSFR is much lower than that of GPSR. This is because links selected by GPSR are not as stable as those in GPSFR. Thus link breakage happens more often in GPSR, requiring data retransmissions that increase delay. Another reason is that there is a smaller ARP delay in GPSFR. For example, if the chosen next hop is not a new one then the ARP request/reply process will not be required because the MAC information of receiver can be retrieved from the cache table. GPSFR prefers to use stable links, which means fewer changes in next hops. This reduces both the ARP delay

and data transmission delay. We also note that shortening the time-out period does not help to reduce the delay because only successfully delivered packets are used for determining delay. Though shorter time-out period can increase the number of successfully delivered packets, it does not reduce the queuing, ARP and transmission delay.

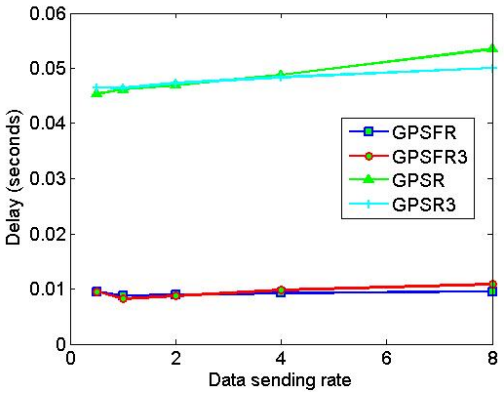


Fig. 9. Network delays various different data sending rate.

E. Effect of route distance

The data delivery ratio of GPSFR and GPSR will decrease as route length increases, as shown in Fig.10. However, delivery ratio of GPSFR is always higher than that of GPSR. For high and low density networks, the performance of both protocols is measured as the distance between the source and destination increases. For high and low density networks, the average distance between vehicles are 50m and 75m, respectively. In all cases, GPSFR has higher delivery ratio than GPSR. Note that GPSFR is also not as sensitive as GPSR to variation of network density.

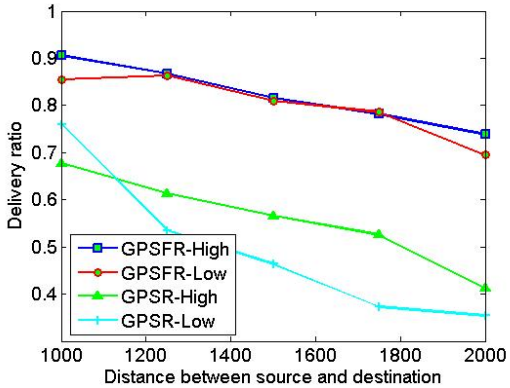


Fig. 10. Data delivery ratio various different routing path length.

VI. CONCLUSION AND FUTURE WORK

The GPS-Free Geographic Routing (GPSFR) algorithm uses relative positions and velocity to achieve higher packet delivery ratio, lower delay and smaller per-node routing state than GPSR, on densely and highly dynamic vehicular networks. Furthermore, it does not require nodes' positions. The *BADV* metric in this geographic routing ensures not only a stable link, but also a higher data delivery ratio and lower delay. Actually, the performance of GPSFR can be further improved if some nodes with GPS locations were added into the network.

Our future work is to design an enhanced protocol based on GPSFR to meet the communication requirement of vehicles in urban areas. While we have shown herein the benefits of GPSFR as a routing protocol for VANET, combining the GPSFR algorithm with a location database system will further reduce the overhead in using external geographic information for routing. An efficient distributed location service would enable the network to be more useful and powerful.

REFERENCES

[1] J. Zhao and G. Cao, "VADD: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks," in *IEEE INFOCOM*, Barcelona, Spain, Apr. 2006

[2] H. Wu, R. Fujimoto, R. Guensler and M. Hunter, "MDDV: a mobility-centric data dissemination algorithm for vehicular networks," in *VANET* '04, Philadelphia, PA, USA, 2004, pp. 47—56

[3] T. Taleb, M. Ochi, A. Jamalipour, N. Kato, and Y. Nemoto, "An efficient vehicle-heading based routing protocol for vanet networks," in Wireless Communications and Networking Conference, 2006. WCNC 2006. IEEE, vol. 4, April 2006, pp. 2199–2204

[4] R. M. Yadumurthy, A. Chimalakonda, M. Sadashivaiah and R. Makanaboyina, "Reliable MAC broadcast protocol in directional and Omni-directional transmissions for vehicular ad hoc networks," in *VANET* '05, Cologne, Germany, 2005, pp. 10—19

[5] M. Sadashivaiah, R. Makanaboyina, B. George and R. Raghavendra, "Performance evaluation of directional MAC protocol for inter-vehicle communication," in *VTC'05-Spring*, 2005 IEEE 61st Volume 4, 30 May-1 June 2005, pp. 2585—2589

[6] T. K. Mak, K. P. Laberteaux and R. Sengupta, "A multi-channel VANET providing concurrent safety and commercial services," in *VANET* '05, Cologne, Germany, 2005, pp. 1—9

[7] H. Füßler, M. Mauve, H. Hartenstein, M. Käsemann, and D. Vollmer, "Poster: Location-Based Routing for Vehicular Ad-Hoc Networks," in *Proc. of ACM MobiCom* '02, Atlanta, Georgia, Sept. 2002.

[8] S. Capkun, M. Hamdi and J.P. Hubaux, "GPS-free positioning in mobile Ad-Hoc networks," in Proc. of Hawaii International Conference on System Sciences, Maui, HW, January 2001, pp. 3481—3490

[9] R. Iyengar and B. Sikdar, "Scalable and Distributed GPS free Positioning for Sensor Networks," in *ICC '03*, IEEE International Conference on Volume 1, 11-15 May 2003, pp. 338—342

[10] A. Rao, C. Papadimitriou, S. Shenker and I. Stoica, "Geographic routing without location information," in *Proc. of ACM MobiCom* '03,San Diego, CA, USA, 2003, pp. 96—108

[11] V. Kukshya, H. Krishnan and C. Kellum, "Design of a System Solution for Relative Positioning of Vehicles Using Vehicle-to-Vehicle Radio Communications During GPS Outages," in *VTC'05-Fall*, 2005 IEEE 62nd Volume 2, 25-28 Sept., 2005, pp. 1313—1317

[12] Report and Press Release – The Economic Impact of Motor Vehicle Crashes, 2000, posted on National Highway Traffic Safety Administration website

[13] C. Liberti and T. S. Rappaport, "Smart Antennas for Wireless Communications: IS-95 and Third Generation CDMA Applications," Prentice Hall, April 1999.

[14] *Geometric Design of Highways and Streets,* American Association of State Highway and Transportation Officials, 2004

[15] F. Borgonovo, L. Campelli, M. Cesana and L. Coletti, "MAC for ad-hoc inter-vehicle network: services and performance," in *VTC'03-Fall*, 2003 IEEE 58th Volume 5, 6-9 Oct. 2003, pp. 2789—2793

[16] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Proc. of ACM MobiCom* '00, Boston, Massachusetts, USA, 2000, pp. 243—254

[17] J. Harri, F. Filali, C. Bonnet, and M. Fiore, "VanetMobiSim: generating realistic mobility patterns for vanets," in VANET '06: Proceedings of the 3rd international workshop on Vehicular ad hoc networks. New York, NY, USA: ACM Press, 2006, pp. 96–97.

Network Modeling of System Goals Attainment

Galia Tzvetkova
Bulgarian Academy of Science
Institute of Mechanics
Email: gtzvet@hotmail.com

Abstract

The aim of this paper is to study problems of analysis, modeling and computer simulations of class discrete event systems. The system under consideration is expected to arrange items in desired manner. Items enter the system area unpredictably and have to be arranged according to a prescribed criterion. The system realizes measurements upon the items, performs processing of the obtained information and takes decisions about items sorting and deployment within its working area.

1. Introduction

The successful deployment of any kind of systems needs an advanced mathematical modeling and simulation. The objective of a modeling process is to evaluate working capabilities of systems and to perform optimization before their physical realization. Based on elaborate mathematical modeling the right simulation model will be built and the appropriate simulation software will be chosen or designed. The expected results from the simulation process are evaluation of working capabilities, estimation of performance parameters and making decisions about overall system optimization.

Diversity of simulation tools, techniques and methods for modeling, analysis and design of systems can be found in the literature. The review of these techniques could be a subject of a special paper.

The preliminary system modeling and optimization considers problems of: analysis of operations that the system will perform, functional systematization and goal-oriented decomposition; definition of general system and sub-system functions and goals, development of a mathematical model, choice of criteria for evaluation, obtaining results and making conclusions from simulations, taking decisions about the overall system optimization.

This paper is devoted to one of the абоже listed problems. It is shown how system functions and goals are realized using Petri nets and network modeling. A system for sorting out and allocation in groups of items is considered.

2. Goal-Oriented Decomposition

The system under consideration operates as follows: A continuous flow of items enters the system. A robot transports each item to a measuring position. The size of the item is measured and a sorting group is assigned for this item. The robot moves the item to a corresponding positions, P_q, q=1,2,…Q.

The functional model is constructed in the following sequence: The first stage is devoted to the systematization of goal-oriented functional information. The result is a statement for general-system functions and general-system goals. For the considered system this statement is: The general-system function is sorting of items under specific dimensions and the general-system goal is the deployment of the sorted items in groups.

Next, the general-system functions and general-system goals are decomposed to functional sub-systems. The functional decomposition results in three sub-systems with their functions and goals defined as follows:

Sub-System "Transportation"
Sub-system function: moving items into defined positions of the working space, sub-system goal: ensuring motion of items flow.

Sub-System "Measurement"
Sub-system function: realization of specifically assigned measurements, sub-system goal: obtaining measurement information.

Sub-System "Sorting"
Sub-system function: processing of the obtained measurement information, sub-system goal: determination of item's sorting group.

The above goal-oriented functional decomposition is used to represent the system by a systemic graph, depicted in Fig.1.

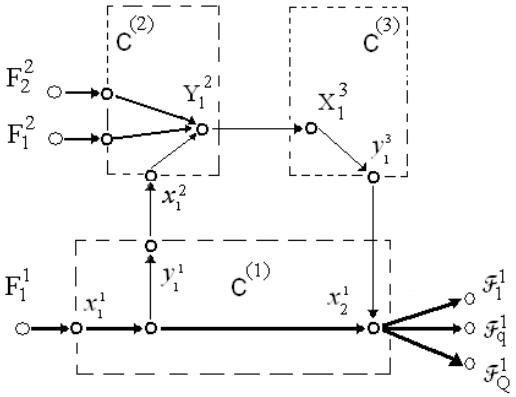


Fig.1. Systemic Graph Model

The systemic graph-model consists of three sub-system models $C^{(1)}$, $C^{(2)}$, $C^{(3)}$ that represent the sub-system functions. The edges of the graph are associated with the exchange of information between the sub-systems. The graph-model variables have the following meaning:

$C^{(1)}$:
- F_1^1 - registration of item within the system range,
- x_1^1 - item moves to distribution position,
- x_2^1 - item moves to deployment position,
- y_1^1 - item arieves at the decision location,
- $\mathcal{F}_1^1, \ldots \mathcal{F}_q^1, \ldots \mathcal{F}_Q^1$ - item arrieves at 'q' location.

$C^{(2)}$:
- F_1^2 - number ofmeasurements N,
- F_2^2 - number ofmeasurements L,
- x_1^2 - begining of measurements,
- Y_1^2 - (NxL) count of measurements.

$C^{(3)}$:
- x_1^3 - (NxL) matrix of measurements,
- y_1^3 - cooorinates of deployment position 'q'.

3. The Simulation Tool

Petri nets are applied for analysis and modeling of the operations of the system. The structure of the functional model is described as [1]:

$$S = (P, T, D^-, D^+, M) = (S^S, M) \qquad (1)$$

where $P = \{P_i\}, i = 1,\ldots n$ is a set of places, $T = \{T_j\}, j = 1,\ldots m$ is a set of transitions, D^- is $(m \times n)$ incidence matrix from places to transitions, D^+ is $(m \times n)$ incidences matrix from transitions to places, M is marking, $S^S = (P, T, D^-, D^+)$ known as structure of the net.

The evaluation of the functions and the structure can be done based on the properties attainability and restrictedness of the net structure. The Petri net (1) is reachable when for a given S^S with marking M^{ini} and another marking M^{fin} is fulfilled $M^{fin} \in R(S^S, M^{ini})$ where R is the set of attainability. If for a given net $S = (S^S, M^{ini})$ a given marking M^{fin} is reachable, then the system attains its goal. The final marking M^{fin} is reachable from the initial marking M^{ini} for a given net $S = (S^S, M^{ini})$ when the matrix equation:

$$M^{fin} = M^{ini} + XD \qquad (2)$$

has a solution in the domain of non-negative integers. The components of the vector $X = [x_1, x_2, \ldots, x_m]^T$ determine the firing sequence of transitions $\{T_j\}$ to reach the marking M^{fin} The sum of the components of the vector X

$$K^{fin} = \sum_{i=1}^{m} x_i \qquad (3)$$

gives the number/count of firings of transitions to reach the marking M^{fin}.

A place P_i of a net $S = (S^S, M^{ini})$ with initial marking M^{ini} is restricted if the condition $M^{fin}(P_i) \le 1$ is fulfilled for all final markings $M^{fin} \in R(S^S, M^{ini})$. The net is restricted if all its places are restricted. The conclusion is: if the net (1) is restricted and reachable for a given final marking M^{fin} than the system represented by this net attains its goal in a finite number of steps.

The marking is considered as a vector $M = [M(P_1) \ldots M(P_n)]^T$. The firing vector $U(k)$ for a state k is m-dimensional {0,1} vector with its j-th component 1 if the transition T_j is allowable for the state k and 0 if not. The firing vector satisfies the equation:

$$(D^-)^T U(k) = M(k) \qquad (4)$$

The main equation that describes the k-th system state is [2]:

$$M(k+1) = M(k) + U(k)^{\mathrm{T}} D$$
$$M(0) = M^{ini}, \quad k = 0,1,2,...,K^{fin} \tag{5}$$

The sequence of transitions shows how the system reaches the designed goal. The system attains its goal if the solution *X* of the equation (2) fulfils the equation (5) for each state *k* and the number of states necessary to attain the goal is given by (3).

The next section shows attainment of the sub-systems functions and sub-system goals of the system.

4. Modeling Attainment of Goals

The modeled system is depicted in Fig.2.

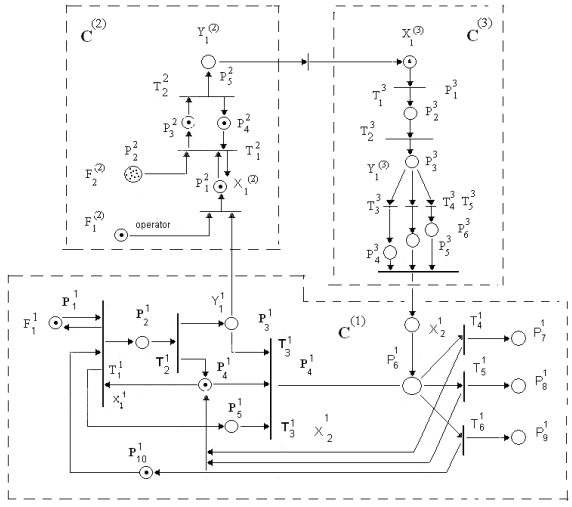


Fig. 2. The Simulation Model

4.1. Sub-System "Items Transportation"

The Petri net that models the deployment is shown in Fig.2, sub-system $C^{(1)}$. The places P_i^1, $i = 1,2,...,10$ have the following meaning: P_1^1 –the item is at the system input (F_1^1); P_2^1 –the item is transported to the measurement position; P_3^1 -the item is on the measurement position (Y_1^1); P_4^1 -robot state; P_5^1 -classification task; P_6^1 -deployment to the corresponding place; P_7^1, P_8^1, P_9^1 -classification groups, P_{10}^1 robot state.

The transitions T_j^1, $j = 1,2,...,6$ have the following meaning: T_1^1 -starts transportation to measurement position (x_1^1), T_2^1 -ends of transportation, T_3^1 -starts transportation from measurement position (x_2^1), T_4^1 - the instant time when the item reaches first classification group $\mathcal{F}_1^1$; T_5^1 -the instant when the item reaches second classification group $\mathcal{F}_2^1$; T_6^1 -the instant when the item reaches third classification group $\mathcal{F}_3^1$;

$$M^{ini} = [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]^{\mathrm{T}} \tag{6}$$

A desired final marking could be:

$$M^{fin} = [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad G_1 \quad G_2 \quad G_3 \quad 1]^{\mathrm{T}} \text{ .....}(7)$$

where G_1, G_2 and G_3 are positive integers representing the numbers of the items belonging to 1-st, 2-nd and 3-rd sorting group.

Let's assume that $G_3 = 3$, $G_2 = 2$, $G_1 = 1$. Than the matrices D^-, D^+ and D are:

$$D^- = \begin{bmatrix} 1&0&0&1&0&0&0&0&0&1 \\ 0&1&0&0&0&0&0&0&0&0 \\ 0&0&1&1&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0 \end{bmatrix} \quad D^+ = \begin{bmatrix} 1&1&0&0&1&0&0&0&0&0 \\ 0&0&1&1&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&1&0&0&1&0&0&1 \\ 0&0&0&1&0&0&0&1&0&1 \\ 0&0&0&1&0&0&0&0&1&1 \end{bmatrix}$$

$$D = \begin{bmatrix} 0&1&0&-1&1&0&0&0&0&-1 \\ 0&-1&1&1&0&0&0&0&0&0 \\ 0&0&-1&-1&-1&1&0&0&0&0 \\ 0&0&0&1&0&-1&1&0&0&1 \\ 0&0&0&1&0&-1&0&1&0&1 \\ 0&0&0&1&0&-1&0&0&1&1 \end{bmatrix} \tag{8}$$

The solution of the matrix equation (2) is $X = [6 \quad 6 \quad 6 \quad 3 \quad 2 \quad 1]^{\mathrm{T}}$. The net depicted in Fig.2 is restricted. The desired final marking is reached for finite number of steps $K^{fin} = 24$. The system equation (5) together with (4),(6),(7) and (8) describes the sub-system states:

$k = 0, U(0) = [1 \ 0 \ 0 \ 0 \ 0 \ 0]^{\mathrm{T}}, M(1) = M(0) + U(0)^{\mathrm{T}} D = [1 \ 1 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0]^{\mathrm{T}}$

$k = 1, U(1) = [1 \ 1 \ 0 \ 0 \ 0 \ 0]^{\mathrm{T}}, M(2) = M(1) + U(1)^{\mathrm{T}} D = [1 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1]^{\mathrm{T}}$

……

.

$k = 22, U(22) = [1 \ 0 \ 0 \ 1 \ 0 \ 0]^{\mathrm{T}}, M(23) = M(22) + U(22)^{\mathrm{T}} D = [1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 3 \ 2 \ 0 \ 1]^{\mathrm{T}}$

$k = 23, U(23) = [1 \ 1 \ 0 \ 0 \ 1 \ 1]^{\mathrm{T}}, M(24) = M(23) + U(23)^{\mathrm{T}} D = [1 \ 0 \ 0 \ 1 \ 0 \ 0 \ 3 \ 2 \ 1 \ 0]^{\mathrm{T}}$

$$M(24) = [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 3 \quad 2 \quad 1 \quad 1]^{\mathrm{T}} = M^{fin} \tag{9}$$

Result: The sub-system "*Items Transportation*" reaches the desired final state M^{fin} in 24 steps. The sub-system attains its sub-system goal, i.e. ensures the movement of the items under classification so that it attains its sub-system goal.

4.2. Sub-System "Measurement"

The Petri net that models the items data acquisition is shown in Fig.2, sub-system $C^{(2)}$. The places P_i^2, $i = 1,2,...,5$ have the following meaning: P_1^2 –the measurement is allowed (x_1^2), P_2^2 – assigned number of measurements, (F_2^2), P_3^2 – realization of a single measurement, P_4^2 –positioning at the next point of measurement, P_5^2 –array of measurement information, (Y_1^2). The transitions T_j^2, $j = 1,2$ are: T_1^2 –beginning of a single measurement; T_2^2 –end of a single measurement.

The initial marking for this sub-net is:

$$M^{ini} = [1 \quad NL \quad 0 \quad 1 \quad 0]^T = [1 \quad 9 \quad 0 \quad 1 \quad 0]^T \qquad (10)$$

And the desired final marking M^{fin} is:

$$M^{fin} = [1 \quad 0 \quad 0 \quad 1 \quad NL]^T = [1 \quad 0 \quad 0 \quad 1 \quad 9]^T \qquad (11)$$

Where NL=9 is assigned number of measurement for an item.

The matrices D^-, D^+ and D for this sub-system are:

$$D^- = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix},\ D^+ = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},\ D = \begin{bmatrix} 0 & -1 & 1 & -1 & 0 \\ 0 & 0 & -1 & 1 & 1 \end{bmatrix} \qquad (12)$$

The solution for the matrix equation (2) of the net in Fig.2, sub-system $C^{(2)}$, is $X = [9 \quad 9]^T$. The net is restricted and the desired final marking (11) is reachable in finite number of steps $K^{fin} = 18$. The system equation (5) together with (4), (10), (11) and (12) describes the sub-system states:

$$k = 0,\ U(0) = [1 \quad 0]^T,\ M(1) = M(0) + U(0)^T D = [1 \quad 8 \quad 1 \quad 0 \quad 0]^T$$

$$k = 1,\ U(1) = [1 \quad 1]^T,\ M(2) = M(1) + U(1)^T D = [1 \quad 7 \quad 1 \quad 0 \quad 1]^T$$

$$k = 2,\ U(2) = [0 \quad 1]^T,\ M(3) = M(2) + U(2)^T D = [1 \quad 7 \quad 0 \quad 1 \quad 2]^T$$

……..……..

$$k = 16,\ U(16) = [0 \quad 1]^T,\ M(17) = M(16) + U(16)^T D = [1 \quad 0 \quad 1 \quad 0 \quad 8]^T$$

$$k = 17,\ U(17) = [0 \quad 0]^T,\ M(18) = M(17) + U(17)^T D = [1 \quad 0 \quad 0 \quad 1 \quad 9]^T$$

$$M(18) = [1 \quad 0 \quad 0 \quad 1 \quad 9]^T = M^{fin} \qquad (13)$$

Result: The marking (13) shows that the desired final state is reached in 18 steps. Therefore, the sub-system "Measurement" reaches its sub-system goal, i.e. obtaining measurement information.

4.3. Sub-System "Sorting"

The Petri net of this sub-system function is shown in Fig.2, sub-system $C^{(3)}$. The places P_i^3, $i = 1,2,...,6$ have the following meaning: P_1^3 –array of measurement information (x_1^3); P_2^3 –processing of the array of measurement information; P_3^3 –procedure to determine the classification group of an item; P_4^3, P_5^3, P_6^3 –sorting groups (q=1,2,3). The transitions T_j^3, $j = 1,2,...,5$ are: T_1^3 –beginning of the measurement information processing, T_2^3 –end of measurement information processing, T_3^3, T_4^3, T_5^3 moments of determining of classification groups.

The initial marking M^{ini} for this sub-net is:

$$M^{ini} = [1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T \qquad (14)$$

and the desired final marking M^{fin} is:

$$M^{fin} = [0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0]^T \qquad (15)$$

The matrices D^-, D^+ and D are:

$$D^- = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad D^+ = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad D = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 \end{bmatrix} \qquad (16)$$

The solution for the matrix equation (2) is $X = [1 \quad 1 \quad 0 \quad 1 \quad 0]^T$ and the system reaches the desired final state for $K^{fin} = 3$ steps. The system equation (5) together with (4), (14), (15) and (16) modeled the sub-system transitions through the states:

$$k = 0,\ U(0) = [1 \quad 0 \quad 0 \quad 0 \quad 0]^T,\ M(1) = M(0) + U(0)^T D = [0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0]^T$$

$$k = 1,\ U(1) = [0 \quad 1 \quad 0 \quad 0 \quad 0]^T,\ M(2) = M(1) + U(1)^T D = [0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0]^T$$

$$k = 2,\ U(2) = [0 \quad 0 \quad 0 \quad 1 \quad 0]^T,\ M(3) = M(2) + U(2) D = [0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0]^T$$

$$M(3) = [0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0]^T = M^{fin}$$

Result: The sub-system "Sorting" attains its sub-system goal, i.e. determines the items sorting group q.

5. Conclusion

A systemic approach is used to represent the functional and the structural organization of discrete event dynamical systems. It is shown that the modeled system attains it functional goals and fulfills its designation. Further efforts will be directed towards study and estimation of quantitative parameters of systems and their optimization.

6. References

[1] Peterson J L. *Petri Net Theory and the Modeling of Systems*, Englewood Cliffs, NJ: Prentice-Hall, 1981.

[2] A. Ichikawa, K. Hiraishi, ”Analysis and control of discrete-event systems represented as Petri nets“, *Lecture Notes in Control and Information Sciences*, No 103, Springer-Verlag, 1988, pp. 115–134.

Estimating Data Redundancy in Sensor Networks

Suman Kumar, S. Srivathsan, Seung-Jong Park, S.S. Iyengar
Sensor Network Research Group,
Computer Science Department
Louisiana State University
Baton Rouge, LA, USA
{sumank, ssrini1, sjpark, iyengar}@csc.lsu.edu

***Abstract*—Large sensor network application includes collection, sharing, processing and storing of data with other sensor devices in the network. Spatio-Temporal correlation of data among sensor devices results in wastage of link bandwidth and device energy. In this paper, a model is constructed using device parameters such as transmission range, sending rate etc., to estimate the data redundancy among closely located sensor devices. We argue that constructing such a model helps in building efficient algorithms for data dissemination in the sensor network and reduce the amount of redundancy in the actual data transfer paths.**

***Index Terms*—Sensor Network, Performance Evaluation, Simulation**

I. Introduction

Technology's rapid growth has resulted in a new class of distributed systems called wireless sensor networks [1] [2]. Wireless sensor networks consist of a large number of small nodes [3]with the capacity to sense, store, process and communicate data wirelessly [4], [5]. Typically, the wireless communication range is small (50 - 100 meters). Due to the improvement in technology, these sensors could be very tiny and yet have all the necessary components in them to perform the intended task autonomously or semi-autonomously. The usefulness of these low-cost, low-power multi-functional nodes is not fully known, but, their usefulness is already being realized in many applications like environmental monitoring, forest fire, chemical/biological plant safety, HVAC systems, defense, civil engineering, border surveillance and many more.

Sensor networks are typically data driven where the whole network cooperates in communicating data from source sensors to sinks (typical repository/server). One of the main characteristics of a typical sensor node is the limited power supply it has [6]. Usually, it is battery operated which might last for some months to a year (depending on the type of application and other application specifications). Sensing nodes typically exhibit limited capabilities in terms of processing, communication, and especially, power [7]. Different application would have different constraints and priorities on how their sensor network must behave. Thus, energy conservation is of prime consideration in sensor network protocols in order to maximize the network's operational lifetime. Rather than sending individual data items from sensors to sinks, it is more energy efficient to send aggregated data. The net effect of this aggregation is, by transmitting less data units, considerable energy savings can be achieved which is the main idea behind in-network [8] aggregation and further distributed processing of the data.

Tiny sensors are often deployed densely to support reliability, fault tolerance etc. This dense deployment causes problems like scalability, redundancy etc. On the one hand, the high density of sensors expends a lot of energy, but on the other hand, it provides much room and many opportunities for us to design energy efficient protocols. This indirectly would lead to the detection of redundant data and minimization of the number of redundant data floating around in the network. However, the heuristic of utilising nodes redundancy has also been used in wireless ad hoc networks [2], [9] and [10]. Hence, data redundancy exploitation is a desired functionality for a sensor network.

In this paper we present and analyze the redundancy problem for wireless sensor networks, although, the work presented in this paper is preliminary and is a step toward redundancy estimation. The analysis and quantification of the level of redundancy would help in the design of QoS-related algorithms which would require such metrics. The remainder of this paper is organized as follows. In

Section II, related work in this area is provided in brief. In Section III, motivation and contribution behind this paper is discussed. In section IV, we introduce our model for redundancy with a probabilistic expression dealing with near neighbor distribution is presented. Finally, the concluding remarks are discussed in Section V.

II. Related Work

In the literature, there has been work on covering a circle by arcs. Early in 1982, Siegel and Holst [11] studied the problem of covering a circle with random arcs of random sizes and established a complicated integral representation for the coverage probability.

Gao *et al.* have estimated the redundant sensing areas and have computed the minimum and maximum number of neighborhood sensors that are required to provide complete redundancy and have also formulated a method to estimate the degree of redundancy without the knowledge of location and directional information [12]. The authors have focussed on the number of redundant nodes in the network as against our focus on the redundant data which could arise irrespective of the presence of redundant nodes. Though the authors have estimated the probability of complete redudancy with tight bounds, they have not directly dealt with the analysis of redundant data in the network and the different probabilities with which the individual sensors would transmit.

Bogdan *et al.* study the problem of detecting and eliminating redundancy in sensor networks with an intention to improve energy efficiency and maintain the coverage of the network [13]. Their work also provides a study of the impact of eliminating redundancy on the coverage-boundary detection. They reduce these problems to the computation of Voronoi diagrams. Our paper is slightly different from this one in the sense that our paper estimates the data redundancy whereas, the paper by Bogdan *et al.* deals with redundant sensors and ways to eliminate them without affecting the coverage area.

Anh *et al.* in [14] analyze the effect of redundancy on the mean time to failure of a network of wireless sensors in terms of energy-depletion. The authors show that there is a tradeoff between the degree of redundancy and the mean time to failure (MTTF). This paper deals with hardware redundancy and path redundancy to deal with sensor faults as against information redundancy and their estimation.

Qilian *et al.* in [15] recongnize the negative effects of data redundancy and propose a Singular-Value-QR Decomposition method to reduce the data redundancy in wireless sensor networks. They first show that adjacent nodes generate similar data and establish the fact the redundancy exists among dense clusters of nodes. To reduce redundancy, they use the SVD method for subset selection and turn off other nodes. This paper is about reducing redundancy and not about estimating how much data redundancy is there in the network.

III. Motivation and Main Contribution

A. Motivation

Densely deployed sensors would result in many sensors that would monitor some mutually common spatial region due to the overlapping of their individual sensing areas. Hence, the data reported by these sources would be correlated. To understand this, in the figure 1, we have sensor A and sensor B and the solid circle drawn around them represent these node's sensing area. As seen in the figure, sensed regions are partially overlapping with each other. When sensor A and sensor B sends data, that data is spatially distributed on their whole sensing region. Hence, the overlapping area is sensed by both of the sensors, resulting in duplicate information. If this data is transferred to the sink, same data set from several sensors are observed by the sink. This results in wastage of transmission energy for sensors and the amount of information provided contains lots of duplicates. Hence, such correlations have to be eliminated as soon as possible. The existence of above mentioned spatial correlation resulting in data redundancy can have potential advantages for the development of efficient communication protocols well-suited for the WSN paradigm. For example, intuitively, due to the spatial correlation, data from spatially separated sensors is more useful to the sink than highly correlated data from nodes in proximity. Therefore, it may not be necessary for every sensor node to transmit its data to the sink; instead, a smaller number of sensor measurements might be adequate to communicate the event features to the sink within a certain reliability level. Similarly,

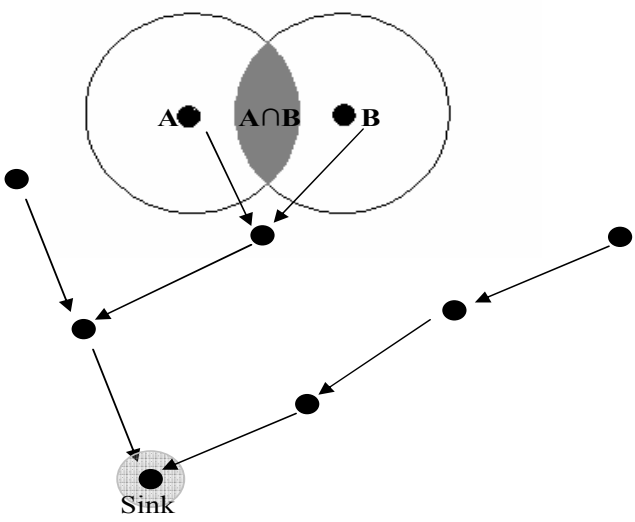


Fig. 1. Sensing Region Overlap Resulting in Data Redundancy

for a certain event-tracking application, the measurement reporting frequency, at which the sensor nodes transmit their observations, can be adjusted such that temporally-correlated phenomena can be captured at the sink within a certain distortion level and with minimum energy-expenditure. To estimate such redundancy phenomena is necessary and is the main aim of this paper.

B. Main Contribution

Although the estimation of redundancy has been studied by simulation [16], to the best of our knowledge, there is no paper presenting a analysis for such an estimation. In this paper, we introduce a heuristic model for data redundancy in spatially distributed sensor network to characterize the amount of redundancy existing among near neighbor nodes. For the general scenario, although in our analysis we introduce two different kind of sensor nodes (further referred as A and B), it does not affect the general analysis for uniform sensor node scenario. However, it may lead to useful result considering that there are at least two kinds of sensor nodes that differ in some sense[1] and still lead to a simplified analysis. In this paper, we consider that whatever differences sensors have, they are distributed with the same master poission process. We recognise that the near neighbor distribution is the main factor contributing to the overlap of sensing regions among nodes that introduces data redundancy among sensor nodes. We give a probabilistic expression giving near node distribution and argue that for a given sensing range how many sensors can deliver partially redundant data.

[1]Sensor Nodes can differ in sensing capability (introduces as factor γ) or those that differ in their sensing cycles which is the subject of future research or they can be entirely different(say optical and magnetic sensors)

IV. System Model

In this section, we continue our discussion from the previous section and introduce our model for data redundancy in sensor network. As discussed earlier spatially overlapped region attracts the joint treatment of data to reduce the communication cost for the data of interest by exploiting the existing correlation. Continuing our two node scenario and assuming data is uniformly distributed throughout the spatial region, the data collected by some node n_i in its sensing region A_i is proportional to the sensing area. Hence, data sensed in area A_i = γA_i Where, γ is some proportionality constant that depends on sensing ability of sensors. Hence, if two sensors A and B sensing the area A_r and B_r transmits the data to the third sensor, meaningful data is given as

$$\gamma(A_r \bigcup B_r) \tag{1}$$

and amount of redundant(duplicate) information is simply given as below assuming the source of redundancy caused by only overlapping sensing region of these two sensors

$$\gamma(A_r \bigcap B_r) \tag{2}$$

One figure of merit is correlation that measures the degree of redundancy which is the percentage of data that is varies with each other and for this case only, overlapping part of the sensing regions varies with each other. Hence, for sensing nodes A and B, the correlation factor is given by equation

$$\frac{A_r \bigcap B_r}{A_r \bigcup B_r} \tag{3}$$

Assuming uniform node configuration of all the nodes, the sensing radius is r_s and transmission range is r_t. the sensing area is given as πr_s^2

For a particular node say s, all the other nodes in area $4{r_s}^2$, shares some degree of redundant information with s. In figure 2, two nodes A and B has position vectors r and r' respectively and r_s is

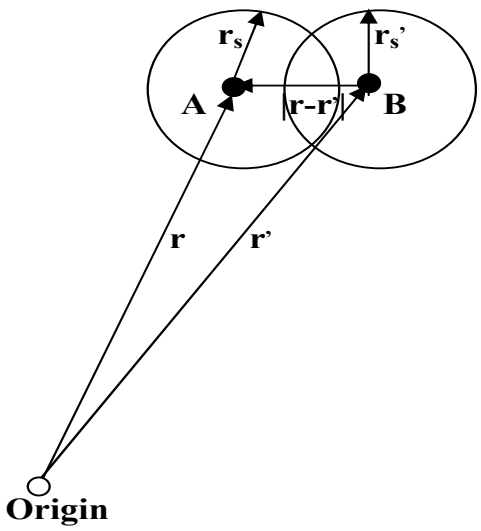


Fig. 2. Redundancy Condition

their sensing range, the condition for redundancy is given by

$$\mid r - r' \mid < 2r_s \tag{4}$$

Hence, to quantify the redundancy for all the neighbors around a sensor node we have to find out its near neighbor distribution in its own sensing range. Our next section presents an analysis, assuming sensor nodes follows a spatial bivariate distribution for sensor nodes A and B. Here, we consider nodes A and B which are different in terms of sensing rate or some other figure of merit say sensing capability factor γ.

A. Nearest Neighbour Distribution

In this section, we present the near neighbor distribution considering the sensor nodes follow a bivariate spatial poisson point pattern [17] and [18] . First, the concept of bivariate distribution is presented in spatial sense and then an expression is derived representing the distribution of the distance from a sensor A to a nearest sensor B.

Maritz [19] obtained the probability generating function for the bivariate poison assuming that, in any interval of length dt, the combinations ($A\overline{B}$, $\overline{A}B$, AB and $\overline{AB}$) of the two events A and B, occur with probabilities λdt, υdt, μdt and $1-(\lambda+\upsilon+\mu)dt$. This model leads to probability generating function

$$G(u,v) = exp[\lambda(u-1) + \mu(v-1) + \upsilon(uv-1)t] \tag{5}$$

The derivation of bivariate poisson distribution shows that the two events A and B are associated in some sense. This association can be described by the conditional probabilities

$$prob(A \mid B) = \frac{\upsilon}{\mu+\upsilon},\ prob(B \mid A) = \frac{\mu}{\mu+\upsilon} \tag{6}$$

We may also consider that we have a process in time and events occur with a poisson density with parameter $(\lambda+\upsilon+\mu)$. Obviously the event is $A\overline{B}$, $\overline{A}B$, or AB with probabilities

$$\frac{\lambda}{\lambda+\upsilon+\mu},\ \frac{\mu}{\lambda+\upsilon+\mu} \text{ and } \frac{\upsilon}{\lambda+\upsilon+\mu} \tag{7}$$

respectively.

Given the probability generation function as above we have two near sensors represting events A and B are associated with the primary poisson process. The association between the two classes of events in this sense is peculiar in the sense that the two events, if they occur they occur together, occur simultaneously.

1) Spatial Bivariate Distribution: From the discussion from the previous section, we can generalize the equation 5 into a spatial distribution by simply replacing the t with an area s. It would then be a probability generating function of the points of type A(sensor of one type) and B(sensor of another type) in an area of size s. We derive a spatial bivariate poisson process with density $\lambda+\upsilon+\mu$. Given a primary point $y = (y_1, y_2)$, with a probability $\frac{\lambda}{\lambda+\upsilon+\mu}$, it gives rise to a point of type A at some nearby point x,. With probability $\frac{\nu}{\lambda+\upsilon+\mu}$, we have a point of type B at x_2 and with probability $\frac{\upsilon}{\lambda+\upsilon+\mu}$, we have a double point with A at x_1, and B at x_2,. The locations x_1, and x_2, are determined so that the (x_1-y) and (x_2-y) are independent and have probability density funotions $f(x_1-y)$ and $g(x_2-y)$, respectively.

Note that the same distribution is obtained from the model where, for a given primary point at y, there occurs a point A or B at y with probabilities $\frac{\lambda}{\lambda+\upsilon+\mu}$ and $\frac{\nu}{\lambda+\upsilon+\mu}$, respectively, or then, with a probability $\frac{\upsilon}{\lambda+\upsilon+\mu}$, a double point with A at x, and B at x,. The distances (x_1-y) and (x_2-y) are independent and have density functions $f(x_1-y)$ and $g(x_2-y)$. It now follows that

prob (a single point A in dx) = λdx,

prob (a single point B in dx) = νdx,

prob (a double point with A in dx, and B in dx)

$= \nu h(x_1, x_2)dx_1dx_2$

where

$$h(x_1, x_2) = \int_{y \in R} f(x_1 - y)g(x_2 - y) \quad (8)$$

and the integration is whole plane. Let $N_1(S_1)$ denote the number of points of type A in an area S_1 and $N_2(S_2)$ the number of points of type B in S_2. Since the marginal distribution of A and B are poisson and also only double points contribute to the covariance, we get from 8 for two different areas S_1 and S_2,

$$cov(N_1(S_1), N_2(S_2)) = \nu \int_{x_1 \in S_1} \int_{x_2 \in S_2} h(x_1, x_2)dx_1dx_2 \quad (9)$$

In the same line, analogous to 5 gives the probability generating function for the spatial bivariate Poisson distribution

$$G(u, v) = exp[(\lambda s + \upsilon(s - s^{'}))(u - 1) + (\mu s + \upsilon(s - s^{'}))(v - 1) + s^{'}\upsilon(uv - 1)] \quad (10)$$

The distribution of the distance between two adjacent points, the nearest neighbor distribution considering marginal distributions are poisson, we get the following relationship

prob(distance from a point B to next nearest point $B < r) = 1 - exp(-(\mu + \upsilon)\pi r^2)$ (11)

and similarly for A. The distribution of the distance from a point A to a nearest point B may be derived as follows:
prob (distance from a point A to nearest point B $>$ r)
= prob (A single) prob (distance from A to nearest B $> r \mid A$ single) + prob (A double) prob (distance from A to nearest B $> r \mid A$ double)

$$= \frac{\lambda}{\lambda + \upsilon} exp(-(\mu + \upsilon)\pi r^2) + \frac{\lambda}{\lambda + \upsilon} exp(-(\mu + \upsilon)\pi r^2) \int_{|x| > r} h(x, 0)dx \quad (12)$$

Hence, prob (distance from a point A to nearest point B $<$ r)=

$$1 - exp(-(\mu + \upsilon)\pi r^2)(\lambda + \upsilon \frac{\int_{|x| > r} h(x, 0)dx}{(\lambda + \upsilon)}) \quad (13)$$

When A and B are independent, i.e. when υ = 0 , 13 reduces to the distribution of the distance from a random point to the nearest point B which is the same distribution as given in 11. When A and B are independent, i.e. when υ = 0 , 13 reduces to the distribution of the distance from a random point to the nearest point B which is the same distribution as given in 12. For the redundancy range $2r_s$ 13 is given as below:

$$1 - exp(-(\mu + \upsilon)\pi 2r_s^2)(\lambda + \upsilon \frac{\int_{|x| > 2r_s} h(x, 0)dx}{(\lambda + \upsilon)}) \quad (14)$$

V. CONCLUSION

Dense deployment of sensor network results into better operation using collaborative nature of wireless sensor networks. This collaboration results in redundant data which proved as a unique characteristic of a typical sensor network. In this paper we introduced a redundancy model. In this model, we observed that redundant data occurs when the nearby sensor devices are separated by not more than twice their sensing radius. It is seen that when condition of redundancy meets near neighbor distribution, which is a very important factor which gives the degree of overlap among sensors in the near neighbor distribution. In this preliminary work, a case of data redundancy is analysed. Sensors are assumed to be distributed and follow a bivariate poisson process. A probabilistic expression is found for near neighbor distribution.

Our results will benefit the research on wireless sensor networks by providing simple estimation of redundancy for designing new energy efficient protocols. However, the present work is in its initial phase. Our further work involves finding the distribution of overlap area and then give an estimation of redundancy using different distributions(including random) of sensor nodes in an event field. Further extension of this work also involves analysis of sensor network for different variation and distribution of sensor nodes. We believe, the presented model may be unrealistic for concrete sensor technology which usually takes into account the sensor orientation, angular aperture, time-varying properties of detections (using different sleep and wake up cycles), etc. In this sense, the analytical results of this paper are only suitable for the situation where each

node can persistently perceive phenomena within its surrounding circular area (as we assumed the uniform sensing radius). Since it is very difficult to provide a general model that can capture the behavior of different sensors for different sensing tasks (say magnetic, optical etc), we are trying to expand our analysis to particular sensor technology and for various sensor network application scenarios.

REFERENCES

[1] Y. Sankarasubramaniam I. F. Akyildiz, W. Su and E. Cyirci, "Wireless sensor networks: A survey," *COMPUTER NETWORKS*, vol. 38, no. 4, pp. 393–422, 2002.

[2] S. Sitharama Iyenger and Richard R. Brooks, *Distributed Sensor Networks*, Chapman and Hall/CRC publishing house, 2005.

[3] R. Katz J. Kahn and K. Pister, "Next century challenges: Mobile networking for smart dust," in *Fifth annual international conference on Mobile computing and networking (MobiCom 99)*, Seattle, USA, 1999, pp. 263–270.

[4] Mohammad Ilyas and Imad Mahgoub, *Handbook of Sensor Networks - Compact Wireless and Wired Sensing Systems*, CRC Press, 2004.

[5] Anna Hac, *Wireless Sensor Network Designs*, John Wiley and Sons, 2003.

[6] R. Katz J. Kahn and K. Pister, "Next century challenges: Scalable coordination in sensor networks," in *Proceedings of International Conference on Mobile Computing and Networking (MobiCom)*, Seattle, USA, 1999.

[7] G.J.Pottie and W.J.Kaiser, "Wireless integrated network sensors," *Communications of the ACM*, vol. 43, no. 5, pp. 51–58, May 2000.

[8] Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," *Proceedings of the 5th symposium on Operating systems design and implementation*, vol. 39, pp. 131–146, 2002.

[9] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," *In Proceedings of the fifth annual international conference on Mobile computing and networking (MobiCom 99)*, pp. 85–96, 1999.

[10] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," *In Proceedings of the seventh annual international conference on Mobile computing and networking (MobiCom 01)*, pp. 70–84, 2001.

[11] A. Siegel and L. Holst, "Covering the circle with random arcs of random sizes," *Journal of Applied Probability*, pp. 19:373–381, 1982.

[12] Fulu Li Yong Gao, Kui Wu, "Analysis on the redundancy of wireless sensor networks," in *Wireless Sensor Networks and Applications*, San Diego, CA, September 19 2003.

[13] Jan Vitek Octavian Carbunar Bogdan Carbunar, Ananth Grama, "Redundancy and coverage detection in sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 2, no. 2, Feb. 2006.

[14] Anh Phan Speer and Ing-Ray Chen, "Effect of redundancy on mean time to failure of wireless sensor networks," in *Proceedings of the 20th International Conference on Advanced Information Networking and Applications (AINA'06)*, Vienna, Austria, April 18-20 2006.

[15] Qilian Liang and Lingming Wang, "Redundancy reduction in wireless sensor networks using svd-qr," in *Military Communications Conference, 2005. MILCOM 2005. IEEE*, Vienna, Austria, Oct 17-20 2005, pp. 1857–1861.

[16] S. Ni, Y. Tseng, Y. Chen, and J. Sheu., "The broadcast storm problem in a mobile ad hoc network," in *MobiCom,*, Aug. 1999, pp. 151–162.

[17] Hui-Hsiung Kuo, *Introduction to Stochastic Integration*, Springer (Universitext), 2006.

[18] Peter J. Diggle, "Spatio-temporal point processes: Methods and application," *Monograph on Statistics and Applied Probability 107, Chapman and Hall/CRC*, pp. 1–45, 2007.

[19] J. S. Maritz, "Note on a certain family of discrete distributions," *Biometrika*, vol. 39, pp. 196–8, 1952.