CONTOUR GUIDED DISSEMINATION
FOR
NETWORKED EMBEDDED SYSTEMS

A Thesis
Presented to
The Graduate Faculty of The University of Akron

In Partial Fulfillment
of the Requirements for the Degree
Master of Science

I-Hsine (Jack) Chu
December, 2006
CONTOUR GUIDED DISSEMINATION
FOR
NETWORKED EMBEDDED SYSTEMS

I-Hsine (Jack) Chu

Thesis

Approved:  

Accepted:  

Advisor  
Dr. Shivakumar Sastry

Department Chair  
Dr. Alex De Abreu Garcia

Committee Member  
Dr. Nathan Ida

Dean of the College  
Dr. George K. Haritos

Committee Member  
Dr. James E. Grover

Dean of the Graduate School  
Dr. George R. Newkome

Date

ii
ABSTRACT

The emergence of new devices and technologies for networked embedded computing motivates new platforms for integrated computation, control, communication and coordination. These platforms are based on a large collection of loosely-coupled microcontrollers that communicate over low-power wireless links to achieve system-level objectives. The low-power transceiver in each node makes it necessary to propagate messages between non-neighboring nodes over multi-hop routes.

Traditional methods for multi-hop communication, on wired networks, rely on unique network identifiers for each node and do not scale well as the number of nodes increases. These methods select one of the available paths between a source and a sink. Extensions to these methods, which are referred to as dissemination methods, have been reported in the sensor networks literature. These extensions are energy efficient and do not rely on network identifiers; however, not all of the available paths between a pair of nodes are fully utilized.

This thesis presents a new approach for disseminating called contour guided dissemination. A contour is the union of all the shortest paths between a pair of nodes. When the network topology is regular, it is shown that a contour can be computed using local computations by each node. It is also shown that, despite the availability of multiple paths, it is not possible to effectively utilize the contour by uniformly spreading messages over the available paths. A heuristic that effectively utilizes the contour and simulation results to demonstrate the effectiveness of the contour guided dissemination is presented.

The results of this thesis are the first steps towards effectively disseminating data in networked embedded systems with general topologies. These results suggest that some nodes must spread the data to exploit the available paths while other nodes must utilize only one of the desirable paths without spreading the data. Identifying
such nodes in systems with general topologies, and integrating contour guided dissemination into system-level protocols are exciting avenues for future research.
DEDICATION

To my wife Crystal and son David
ACKNOWLEDGMENTS

I would like to thank Dr. Sastry for his guidance and patience. Without him, this thesis could not be done. I learned a lot about life from him; he is not only a skilled professor, with technical expertise, but also an educator. It was a fun ride for me to transform myself from a student with Mechanical Engineering background to a graduate in Computer Engineering. Along the way, I was very lucky to have excellent professors teach me in every field of knowledge. Thanks for Dr. Ida and Dr. Grover’s classes, I accumulated most of my knowledge about computer hardware. Thanks to Dr. Carletta for showing me the VLSI manufacturing process and FPGA capabilities. I would also like to thank the Department of Electrical and Computer Engineering at the University of Akron for their teaching assistantship support.

Last, but not least. I would also like to express my thanks to my colleagues: Nunzio, Steve and Ruan. They helped me a lot with the syntax in the programming and debugging processes.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF TABLES</td>
<td></td>
<td>ix</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I.</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
<td>Research Objective</td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>Contributions</td>
</tr>
<tr>
<td></td>
<td>1.3</td>
<td>Overview of the Thesis</td>
</tr>
<tr>
<td>II.</td>
<td>BACKGROUND</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>2.1</td>
<td>Dissemination Protocols</td>
</tr>
<tr>
<td></td>
<td>2.2</td>
<td>Coupled Conveyors</td>
</tr>
<tr>
<td></td>
<td>2.2.1</td>
<td>Example 1: Inspection Station</td>
</tr>
<tr>
<td></td>
<td>2.2.2</td>
<td>Example 2: Package Sorting System</td>
</tr>
<tr>
<td></td>
<td>2.3</td>
<td>The Monitoring Problem</td>
</tr>
<tr>
<td></td>
<td>2.4</td>
<td>Challenges for Monitoring</td>
</tr>
<tr>
<td>III.</td>
<td>CONTOUR GUIDED DISSEMINATION</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>3.1</td>
<td>Contour of Shortest Paths</td>
</tr>
<tr>
<td></td>
<td>3.2</td>
<td>Uniform spreading</td>
</tr>
<tr>
<td></td>
<td>3.3</td>
<td>Orientations of a Contour</td>
</tr>
<tr>
<td></td>
<td>3.4</td>
<td>Heuristic Spreading</td>
</tr>
<tr>
<td></td>
<td>3.5</td>
<td>Summary</td>
</tr>
<tr>
<td>IV.</td>
<td>SIMULATION RESULTS</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>4.1</td>
<td>Simulation Approach and Tools</td>
</tr>
</tbody>
</table>
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Protocols Comparison</td>
<td>7</td>
</tr>
<tr>
<td>2.2 Multi-hop Delivery Scenario 1</td>
<td>12</td>
</tr>
<tr>
<td>2.3 Multi-hop Delivery Scenario 2</td>
<td>13</td>
</tr>
<tr>
<td>3.1 Orientation Conditions</td>
<td>25</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Hierarchical Approach for Routing</td>
<td>6</td>
</tr>
<tr>
<td>2.2 An Inspection System</td>
<td>9</td>
</tr>
<tr>
<td>2.3 A Package Sorter</td>
<td>10</td>
</tr>
<tr>
<td>2.4 Monitoring in Automation Systems</td>
<td>11</td>
</tr>
<tr>
<td>2.5 Monitoring an Networked Embedded System</td>
<td>12</td>
</tr>
<tr>
<td>2.6 Performance of CSMA</td>
<td>13</td>
</tr>
<tr>
<td>3.1 Eight Neighbors of a node $n_{i,j}$</td>
<td>16</td>
</tr>
<tr>
<td>3.2 Layered Paths when $x_{diff}$ is even</td>
<td>17</td>
</tr>
<tr>
<td>3.3 Layered Paths when $x_{diff}$ is odd</td>
<td>18</td>
</tr>
<tr>
<td>3.4 Contour when $xy_{diff}$ is even</td>
<td>20</td>
</tr>
<tr>
<td>3.5 Uniform Spreading</td>
<td>22</td>
</tr>
<tr>
<td>3.6 Orientations of a Contour</td>
<td>23</td>
</tr>
<tr>
<td>3.7 Eight possible Orientations for Contours</td>
<td>24</td>
</tr>
<tr>
<td>3.8 Ideal Messages Loading in the Contour</td>
<td>26</td>
</tr>
<tr>
<td>3.9 Three Regions of Contour</td>
<td>27</td>
</tr>
<tr>
<td>3.10 Heuristic Spreading in Expansion</td>
<td>27</td>
</tr>
<tr>
<td>3.11 Heuristic Spreading in Contraction Region</td>
<td>28</td>
</tr>
<tr>
<td>3.12 Heuristic Spreading in Transition Region</td>
<td>29</td>
</tr>
<tr>
<td>3.13 Heuristic Spreading in Transition Region</td>
<td>29</td>
</tr>
<tr>
<td>4.1 Contour Guided Dissemination Application</td>
<td>31</td>
</tr>
<tr>
<td>4.2 Flooding after 30 (Left) and 60 (Right) Seconds</td>
<td>33</td>
</tr>
</tbody>
</table>
4.3 Flooding after 90 (Left) and 120 (Right) Seconds ........................... 33
4.4 Gossiping after 30 (Left) and 60 (Right) Seconds ......................... 35
4.5 Gossiping after 90 (Left) and 120 (Right) Seconds ..................... 35
4.6 Contour Guided Dissemination - Uniform Spreading in Hexagon (Top)
   and Rectangle (Bottom) .......................................................... 36
4.7 Contour Guided Dissemination - Weighted Spreading in Hexagon (Top)
   and Rectangle (Bottom) .......................................................... 37
4.8 Contour Guided Dissemination - Heuristic Spreading in Hexagon (Top)
   and Rectangle (Bottom) .......................................................... 38
4.9 PowerTOSSIM - Heuristic Spreading in Hexagon (Top) and Rectangle
   (Bottom) ................................................................................. 40
4.10 Scalability Comparison ............................................................... 41
CHAPTER I

INTRODUCTION

Networked embedded systems have emerged as a platform with tremendous capabilities and opportunities for future engineered systems. Such a platform comprises a large number of tiny, resource-constrained microcontroller nodes that have limited capabilities for computation and communication [1]. Sensor networks [2] and networked sensor-actuator systems (NSAS) [3] are two examples of networked embedded systems.

The problem of disseminating messages in networked embedded systems is particularly challenging because of the resource constraints at the node-level, limited bandwidth, high bit-error rates and irregular propagation of the wireless links [4]. 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 [5]. This investigation considered the problem of disseminating the messages from a source node to a sink node when the nodes of a networked embedded system are located on a two-dimensional grid. A new method for dissemination that exploits the location of each node and multiple paths between nodes is presented.

Methods to disseminate messages from a source to one or more sink nodes have been addressed in the networking literature. Traditionally, routing protocols that are based on shortest-path algorithms such as the Bellman-Ford algorithm [6] and Dijkstra’s algorithm [7] are the basis for these methods. For example, distance
vector routing and link-state routing are two well-known protocols that are used in distributed systems [8, 9]. The distance vector routing protocol, which is based on the Bellman-Ford algorithm, requires each node to maintain a routing table. Each entry in the routing table of a node identifies the next node to which a packet destined for a sink, must be sent. If $d(i, snk)$ represents the distance from node $n_i$ to the sink node, and $n_j$ is a neighbor of $n_i$, then the routing table of $n_i$ is updated for a neighbor $n_j$ whenever $d(i, snk) > d(j, snk) + d(i, j)$, i.e., whenever a node discovers a new, shorter, route to the sink. In order to maintain accuracy of the routing tables, each node must notify its neighbors whenever its own routing table changes.

In addition to the message overheads, approaches based on the above algorithms are not suitable for sensor networks and networked embedded system. For example, when the topology is disrupted because of faults, it takes a long time for the routing tables in each node to converge to a stable state. The node resources are constrained and hence, it is not desirable to store and maintain routing tables. When there are multiple monitoring stations, every node must maintain a routing table for every 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 must have more resources than other nodes in the networked embedded system.

1.1 Research Objective

In many engineered systems, the nodes are located at well-designed, fixed, locations. For example, in automotive assembly lines or warehouse conveyor systems, every module is located in a zone. Further, to ensure that the system is robust, multiple paths for communication between a pair of nodes are designed into the system. The motivating question for this investigation was whether all the available paths between a pair of nodes can be effectively used to disseminate messages.
1.2 Contributions

The contributions of this thesis are:

1. A new approach for dissemination.

2. A method to compute the contour using relative locations of the source and the sink.

3. A new heuristic that effectively utilizes the multiple paths in a contour.

1.3 Overview of the Thesis

Chapter II describes the monitoring problem, which is one application for Contour Guided Dissemination. Chapter III presents the Contour Guided Dissemination method. After defining the contour as the union of all shortest paths between a pair of nodes, it is shown that the relative locations of nodes are necessary and sufficient to allow each node to determine its position in a contour. Next, it is shown that to benefit from the multiple paths in a contour, it is not sufficient to uniformly spread the messages among the available paths; surprisingly, one path in the contour gets loaded with more messages than the other paths. A heuristic that efficiently spreads the messages over the available paths is presented. Chapter IV presents the simulation results and Chapter V presents the conclusions of this investigation.
CHAPTER II

BACKGROUND

The problem of sending messages from a source over a multi-hop path is related to routing in wired and wireless networks [10] and to data dissemination in sensor networks [2]. These approaches aim to move data from one or more sources to one or more sinks over a multi-hop network.

The two dominant methods used in wired networks are the distance vector scheme and the link state scheme that are based on shortest path algorithms by Bellman [6] and Dijkstra [7], respectively, as discussed briefly in Chapter I. Contour Guided Dissemination departs from the application-specific methods advocated in the sensor networks literature [5]. Networked embedded systems have unique characteristics such as scale, constrained resources, unreliable channel and unpredictable failures. It is necessary to consider that the execution is periodic and may be based either on an event-triggered or time-triggered model. Nodes are not deployed in an ad-hoc manner and the geometric structure of the nodes can be exploited. The “query-response” model [11] is useful for networked embedded systems — however, the “queries” are not ad hoc; in contrast, they are highly structured; queries are issued infrequently in response to faults that occur in the environment.

2.1 Dissemination Protocols

Flooding and Gossiping [12] are two data-centric schemes for propagating data in multi-hop networks. These schemes are called data-centric because the propagation is not based on the address, or any other identifier, of the nodes involved in the
propagation. In flooding, a node forwards its data to all its neighbors; note, that in a wireless network, this effect can be achieved by a single broadcast. In Gossiping, a node forwards data to a neighbor that is selected at random; once again, note that since broadcast is the primary mode of communication in a wireless medium, the neighboring nodes that are “not-selected” must drop the data. A protocol called SPIN [9] uses meta-data to reduce the data traffic in a sensor network. In this approach, a node advertises the data it has and only sends the data only if a neighbor requests the data.

Reverse-path forwarding [5] is a general approach that is used in several dissemination protocols. In this approach, a sink node (such as a Monitoring Station) generates a “query” that indicates an interest in data items from one or more nodes. Such a query is forwarded along one or more paths from the sink to all the nodes. The nodes forward data along these paths; since the data propagates in a direction that is reverse of the query direction, this method is referred to as reverse-path forwarding. Declarative routing protocol (DRP [13]), Directed Diffusion (DD [14]) and resilient multiple path Directed Diffusion [15] are examples that are based in reverse-path forwarding.

Cost field approach is another choice for routing. Every node maintains a scalar that represents the cost of sending data back to the sink and the data is propagating along the cost gradients. Akkaya and Younis paper [11] divides the cost field approach into two categories: the receiver-decided protocols and the sender-appointed protocols. Protocols based in cost-field require less memory space compared to reverse-path forwarding since there is no need to maintain a routing table. Routing decisions are based in the cost at each node. Since nodes are resource constrained in both communication and computation capabilities, cost field based approach is a good choice.

GRAdient routing in Ad Hoc network (GRAd), simple cost field protocol [16], and GRAdient Broadcast (GRAB [17]) are examples of receiver-decided protocol. First, the sink floods the network with the request message and sets up scalar values. For example: if a node receives from a node which has \( N \) hops to the sink, then it will set up its own scalar as \( N + 1 \) hops to the sink. Energy Aware Routing (EAR [18]) and
Contour Guide Dissemination (CGD) discussed in Chapter III are example of sender-appointed protocols. The sender computes the cost of sending the data along the multiple paths depending on the location of the sink, and then chooses the least cost route to send its messages back to the sink.

**Virtual hierarchy approach:** This approach achieves scalability by applying multiple levels in the network. As shown in Figure 2.1, the nodes form local clusters and a cluster head will be elected from the cluster. Nodes in the cluster may rotate their roles as the cluster head in order to achieve the energy consumption balance in the cluster. Since the radio transmission strength is adjustable, the cluster head node can be set to a more powerful transmission range to reach the monitoring station. Low-energy adaptive clustering hierarchy (LEACH [19]) and Adaptive Threshold sensitive Energy Efficient sensor Network (APTEEN [20]) are examples of such protocols.

![Hierarchical Approach for Routing](image)

Figure 2.1: Hierarchical Approach for Routing

**Geographical forwarding:** By utilizing the GPS or the location on a grid, the network uses location information to achieve a more efficient routing in the network. Geographic adaptive fidelity (GAF [21]) sets up a virtual grid depending on the GPS providing location and saves energy in the network by propagating messages in different nodes in the same region. Geographic and Energy Aware Routing (GEAR [22])
Table 2.1: Protocols Comparison

<table>
<thead>
<tr>
<th>Reverse Path Forwarding</th>
<th>Cost field based approach</th>
<th>Single path</th>
<th>Multiple Path</th>
<th>Add-on features</th>
</tr>
</thead>
<tbody>
<tr>
<td>FWDE</td>
<td>Sender appointed</td>
<td>Receiver decided</td>
<td>x balancing</td>
<td>x balancing</td>
</tr>
<tr>
<td>Gossiping</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DD</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>DRP</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAd</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAB</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EAR</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CGD-1 UDS</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>CGD-2 WDS</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>CGD-3 HDS</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>LEACH</td>
<td>Add-on</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPIN</td>
<td>Add-on</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GEAR</td>
<td>Add-on</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAP</td>
<td>Add-on</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPEED</td>
<td>Add-on</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>APTEEN</td>
<td>Add-on</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GAF</td>
<td>Add-on</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAR</td>
<td>Add-on</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
reduces traffic load by first sending a beacon in a sub region which the sink is interested in and then disseminate the query into this region.

**Real-time delivery and QoS:** the protocols in the category consider the useful life time of a packet and try to send the message back to the sink before the useful life time expires. RAP [23] achieves this by adjusting the packet priority in the queue, back-off and waiting parameters in the MAC layer. SPEED [24] calculates the distance, $D$, between the node with a message and the sink and dividing by the life time, $t$. After getting the velocity $v = \frac{D}{t}$, the message is forwarded to the sink.

**In network data-processing:** Instead of sending all the measured data back to the sink, the local nodes will process the data and only sends out a summary to the sink. Sensor Protocols for Information via Negotiation (SPIN [25]) and directed diffusion are belonged to this field. No matter in what kind of network, reducing the redundant traffic in the network is always preferred.

Table 2.1 shows a summary of the dissemination protocols discussed in this section. Add-on features such as QoS, virtual hierarchy etc, can be integrated into both reverse-path forwarding and cost field approach.

### 2.2 Coupled Conveyors

**Coupled Conveyors** is a method for building conveyor systems with an integrated networked embedded system that regulates its operations. The conveyor systems are assembled using three building blocks called Segment, Turnaround and Crossover [3]. The following are two examples from [3].

#### 2.2.1 Example 1: Inspection Station

Consider the instance of coupled conveyors shown in Figure 2.2. This instance represents an application of coupled conveyors for material inspection and/or sorting. There are 18 Segments, 4 Turnarounds and one Crossover in this instance. The module identities are shown next to each module. For example, port D of Turnaround $T_4$ is embedded at grid location (1,1), and port A of Turnaround $T_2$ is at grid location.
Figure 2.2: An Inspection System

(5,3). Crossover $C_1$ is at grid location (3,3); HCI devices $H_1$, $H_2$, and $H_3$ are at grid locations (1,0), (3,4) and (5,0), respectively. All the other modules are embedded with appropriate directions as shown in Figure 2.2\(^1\).

Entities come in to the system via $S_6$ (at port A of $T_2$) or $S_{17}$ (port A of $T_3$). Entities leave the system via port D of $T_4$ or via $S_1$ (at port D of $T_1$). Entities coming in via $S_6$ may be moved either along the path

$$T_2, S_7, S_8, S_9, S_{10}, T_1, S_1$$

or along the path

$$T_2, S_{16}, T_3, S_{14}, S_{13}, T_4, S_{11}, T_1, S_1.$$  

Entities may be deflected for re-inspection via port C of $T_3$. Defective or oversized material that are likely to damage the inspection devices along the subpath $S_{10}, T_1$ may be diverted using $C_1$. Entities coming into $T_3$ can be moved along the subpath

\(^1\)In addition to the sensors and actuators on the modules such an application would use additional devices such as cameras, RFID tags and scanners, weighing scales, etc. to inspect the entities that are being moved.
Figure 2.3: A Package Sorter

$T_3, S_{14}, S_{15}$, or deflected at port C of $T_3$ for manual inspection. Manually inspected entities may be re-inserted into the system via port B of $T_4$. The HCI devices are used to monitor the status of all modules and the operator panels can be used to issue commands such as start, stop, or force manual inspection for all entities over a duration of time. There are several other possibilities for moving entities in coupled conveyors that are not discussed in the examples in this paper.

2.2.2 Example 2: Package Sorting System

Figure 2.3 shows how coupled conveyors can be used in a package sorting application. There are 22 Segments, eight Turnarounds and one Crossover. Although some Turnarounds such as $T_1, T_2, T_3$ and $T_8$ appear to be different from other Turnarounds, all the Turnarounds have two input ports and two output ports. The behavior of these Turnarounds are not different from the behavior of other Turnarounds. As an example, packages arrive at a warehouse from five input docks, $I_1, I_2, \ldots, I_5$. The objective of the application is to sort the packages into three categories - Small, Medium and Large. As the packages move on the blocks, the part deflectors on Turnaround modules move
to route the packages to different destinations.

A large package that arrived at dock \( I_5 \) would move along the path

\[ S_2, T_1, S_6, T_2, S_{13}, S_{15}, T_6, S_{21}, S_{22}, T_8. \]

Similarly a small packet arriving at dock \( I_2 \) would move along the path

\[ S_{18}, T_5, S_20, T_6, S_{16}, S_{14}, T_3, S_{10}, T_4. \]

### 2.3 The Monitoring Problem

Consider a typical engineered system that is depicted in Figure 2.4. All the operations of this system are regulated by controllers and monitoring stations are directly connected to these controllers. Periodically, the controllers send a block of words to the monitoring stations to update status. Systems software executing in the monitoring stations translate such a block of words to graphical widgets that are presented to users.

Consider an networked embedded system as shown in Figure 2.5. In this system StatusMessages are delivered to monitoring stations over multi-hop routes. Consider four nodes \( n_{1,1}, n_{1,2}, n_{1,3} \) and \( n_{1,4} \) and a monitoring station \( M_1 \). Only node \( n_{1,1} \) can
communicate with M1 in one hop. Let $m_{i,j}^k$ denote the status message generated by segment $n_{i,j}$ in time tick $k$. The multi-hop delivery of StatusMessages in this scenario is shown in Table 2.2.

Table 2.2: Multi-hop Delivery Scenario 1

<table>
<thead>
<tr>
<th>Time</th>
<th>M1</th>
<th>$n_{1,1}$</th>
<th>$n_{1,2}$</th>
<th>$n_{1,3}$</th>
<th>$n_{1,4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>$m_{1,1}^1$</td>
<td>$m_{1,2}^1$</td>
<td>$m_{1,3}^1$</td>
<td>$m_{1,4}^1$</td>
</tr>
<tr>
<td>2</td>
<td>$m_{1,1}^2$</td>
<td>$m_{1,2}^2$</td>
<td>$m_{1,3}^2$</td>
<td>$m_{1,4}^2$</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>$m_{1,1}^3m_{1,2}^3$</td>
<td>$m_{1,3}^3m_{1,4}^3$</td>
<td>$m_{1,3}^4m_{1,3}^4$</td>
<td>$m_{1,4}^4m_{1,4}^4$</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>$m_{1,1}^4m_{1,2}^4m_{1,3}^4$</td>
<td>$m_{1,3}^4m_{1,2}^4m_{1,1}^4$</td>
<td>$m_{1,3}^4m_{1,3}^4m_{1,2}^4$</td>
<td>$m_{1,4}^4m_{1,3}^4m_{1,4}^4$</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>$m_{1,1}^5m_{1,2}^5m_{1,3}^5m_{1,4}^5$</td>
<td>$m_{1,3}^5m_{1,2}^5m_{1,1}^5$</td>
<td>$m_{1,3}^4m_{1,3}^4m_{1,2}^4$</td>
<td>$m_{1,4}^4m_{1,3}^4m_{1,4}^4$</td>
<td></td>
</tr>
</tbody>
</table>

To emphasize that location of monitoring station impacts communication between nodes, consider another scenario in which a monitoring station is located at grid location (1,2). The delivery of StatusMessages over multi-hop routes is shown in Table 2.3. Generalizing the phenomenon in Table 2.2 and 2.3, if

$$P = (v_1, v_2, \cdots, v_n) = M_i$$

is a longest path in the networked embedded system that terminates in some monitoring station, node $v_i$ in $P$ must maintain a buffer of size $n - i$ to store and forward StatusMessages from its preceding nodes in the path.

2.4 Challenges for Monitoring

Networked embedded system nodes are not energy-constrained. However, these nodes are severely resource constrained. These constraints are desirable to ensure
easy scalability and maintenance. Consequently, large buffers cannot be maintained at each node. In addition, the node can only perform limited computation and communication. When the computation load is increased, the capacity for communication, sensing, and actuation is decreased. Also, communication rates cannot be increased because of the performance of the Media Access Control (MAC) protocol.

Carrier Sensing Multiple Access (CSMA) with Collision Avoidance is a basic MAC protocol that is used in sensor networks. Figure 2.6 shows the performance of CSMA as the number of nodes contending for the channel increases and as the rate at which nodes initiate communication messages is increased. It can be noticed that
when the rate increases, the number of messages received without collisions significantly drops. The number of messages received also decreases when the number of nodes contending for the channel increases.

To address these challenges, the approach used here aimed to exploit multiple paths that are available between a pair of nodes.
CHAPTER III
CONTOUR GUIDED DISSEMINATION

This chapter defines a contour as the union of all shortest paths between a pair of nodes. Every node is either on the contour or not; only nodes on the contour participate in the dissemination. Section 3.1 develops results that are sufficient to allow each node to compute its own position on a contour using local computations. It was intuitively appealing to utilize the multiple paths in a contour by uniformly spreading the messages over the available paths. It is shown that this intuitive idea results in one path being loaded, despite the availability of multiple paths. A heuristic spreading method that effectively uses the multiple paths is presented.

3.1 Contour of Shortest Paths

Consider an networked embedded system in which each node $n_{i,j}$ is located at on a grid. Each node can communicate directly with its eight immediate neighbors\(^1\) as shown in Figure 3.1. Because such communication is possible, there exists multiple shortest paths between any a pair of nodes, $n_{i,j}$ and $n_{q,r}$, in the grid.

**Definition 1.** The contour of shortest paths between $n_{i,j}$ and $n_{q,r}$ ("contour of $n_{i,j}$ and $n_{q,r}$" for short) is the union of all the shortest paths between $n_{i,j}$ and $n_{q,r}$. Each path in the contour is different from every other path by at least one edge.

\(^1\)Such communication can be achieved by designing a suitable Medium Access Control protocol, using low-power steerable antennas, or multiple frequencies.
Definition 2. Given a pair of nodes \( n_{i,j} \) and \( n_{q,r} \),

\[
x_{\text{diff}}(n_{i,j}, n_{q,r}) = |q - i|,
\]

\[
y_{\text{diff}}(n_{i,j}, n_{q,r}) = |r - j|,
\]

\[
xy_{\text{diff}}(n_{i,j}, n_{q,r}) = |x_{\text{diff}}(n_{i,j}, n_{q,r}) - y_{\text{diff}}(n_{i,j}, n_{q,r})| , \text{and} \]

\[
\kappa(n_{i,j}, n_{q,r}) = \left\lfloor \frac{xy_{\text{diff}}}{2} \right\rfloor .
\]

When the context is clear, \( x_{\text{diff}}, y_{\text{diff}}, xy_{\text{diff}} \) and \( \kappa \) are used without explicitly specifying the nodes.

Observation 1. The length of the shortest path between nodes \( n_{i,j} \) and \( n_{q,r} \) is 
\[
\max(x_{\text{diff}}, y_{\text{diff}}).
\]

Proof. Follows from the structure of the grid.

Lemma 1. Let \( y_{\text{diff}} = 0 \). Given a pair of nodes \( n_{i,j} \) and \( n_{q,r} \), the contour of \( n_{i,j} \) and \( n_{q,r} \) is a rectangle, with corner nodes \( n_{i,j}, n_{i+\kappa,j-k}, n_{q-k,j+k} \) and \( n_{q,r} \), if and only if \( x_{\text{diff}} \) is even.
Figure 3.2: Layered Paths when $x_{diff}$ is even

**Proof.** First, given $y_{diff} = 0$ and $x_{diff}$ is even, it is shown that the four corner nodes of the contour are $n_{i,j}$, $n_{i+\kappa,j-\kappa}$, $n_{q-\kappa,j+\kappa}$ and $n_{q,j}$. Consider the shortest paths shown in Figure 3.2. The obvious shortest path between the nodes $n_{i,j}$ and $n_{q,j}$ is along the path $< n_{i,j}, n_{i+1,j}, \ldots, n_{q-1,j}, n_{q,j} = r >$. Since $x_{diff}$ is even, there are an odd number of nodes in the path. Let us call this path a Layer 0 path for convenience. As shown in Figure 3.2, there are additional shortest paths between the nodes $n_{i,j}$ and $n_{q,r}$. Any subpath $< \ldots n_{s,t}, n_{s+1,t}, n_{s+2,t} \ldots >$ can be replaced by a path $< \ldots n_{s,t}, n_{s+1,t+1}, n_{s+2,t} \ldots >$ or $< \ldots n_{s,t}, n_{s+1,t\cdot-1}, n_{s+2,t} \ldots >$ without increasing the length of the shortest path. However, in Layer 1, the node $n_{i,j+1}$ cannot be on any shortest path because any path between $n_{i,j}$ and $n_{q,j}$ that passes through $n_{i,j+1}$ must be of length $\geq x_{diff} + 1$. Similarly, node $n_{q,j+1}$ (in Layer 1) cannot be on any shortest path.

It may be noticed that in each layer, $p$, $1 \leq p \leq \kappa$, the number of nodes that can appear on any shortest path between $n_{i,j}$ and $n_{q,j}$ is decreased by $2 \times p$. Since Layer 0 has an odd number of nodes, Layer $\kappa$ must have a single node, i.e., $n_{i+\kappa,j+\kappa}$ which is one corner node. Using similar arguments, it is noticed that there are additional shortest paths between $n_{i,j}$ and $n_{q,j}$ that go through nodes in Layers $-1, -2, \ldots, -\kappa$. Once again, Layer $-\kappa$ must have a single node, i.e., $n_{i+\kappa,j-\kappa}$. Since $x_{diff}$ is even and
$y_{\text{diff}} = 0$, $q - \kappa = i + \kappa$, the four corner nodes of the rectangle are specified in the Lemma.

Next, given $y_{\text{diff}} = 0$ and a contour with four corner nodes $n_{i,j}$, $n_{i+\kappa,j-\kappa}$, $n_{q-\kappa,j+\kappa}$ and $n_{q,j}$, it is shown that $x_{\text{diff}}$ is even.

Consider the path from $n_{i,j}$ to $n_{i+\kappa,j+\kappa}$. The length of this path is the same as the length of the path from $n_{i,j}$ to $n_{i+\kappa,j}$ because of the structure of the grid. It follows from the construction that $x_{\text{diff}}$ is $2 \times \kappa$ and hence must be even.

**Lemma 2.** Let $x_{\text{diff}} = 0$. Given a pair of nodes $n_{i,j}$ and $n_{q,r}$, the contour of $n_{i,j}$ and $n_{i,r}$ is a rectangle, with corner nodes $n_{i,j}$, $n_{i-\kappa,j+\kappa}$, $n_{i+\kappa,j+\kappa}$ and $n_{i,r}$, if and only if $y_{\text{diff}}$ is even.

**Proof.** Similar to the proof of Lemma 1, except that the Layer 0 path will be along the y axis.

$$\hfill$$

**Figure 3.3:** Layered Paths when $x_{\text{diff}}$ is odd

**Lemma 3.** Let $y_{\text{diff}} = 0$. Given a pair of nodes $n_{i,j}$ and $n_{q,r}$, the contour of $n_{i,j}$ and $n_{q,j}$ is a hexagon, with corner nodes $n_{i,j}$, $n_{i+\kappa,j-\kappa}$, $n_{i+\kappa+1,j-\kappa}$, $n_{q-\kappa,j+\kappa}$, $n_{q-\kappa-1,j+\kappa}$ and $n_{q,j}$, if and only if $x_{\text{diff}}$ is odd and $x_{\text{diff}} \geq 3$. 

18
Proof. Consider the shortest paths between node $n_{i,j}$ and $n_{q,j}$, when $y_{\text{diff}} = 0$ and $x_{\text{diff}}$ is odd, as shown in Figure 3.3. The arguments are similar to the proof of Lemma 1. Because $x_{\text{diff}}$ is odd, there are an even number of nodes along the path involving only the nodes in Layer 0. As in the case of Lemma 1, the number of nodes that can be on any shortest path in Layer $p$, $1 \leq p \leq \kappa$, decreases by $2 \times p$ compared to the number of nodes in Layer 0. However, since the Layer 0 path has an even number of nodes, Layer $\kappa$ must have two nodes, i.e., $n_{i+\kappa,j+\kappa}$ and $n_{i+\kappa+1,j+\kappa}$. Similarly, Layer $-\kappa$ will have the two nodes $n_{i,j-\kappa}$ and $n_{i+1,j-\kappa}$.

Given $y_{\text{diff}} = 0$ and a contour with six corner nodes $n_{i,j}$, $n_{i+\kappa,j-\kappa}$, $n_{i+\kappa,j+\kappa}$, $n_{q-j+\kappa}$, $n_{q-j-\kappa}$, and $n_{q,j}$, consider the path from $n_{i,j}$ to $n_{i+\kappa,j+\kappa}$. The length of this path is the same as the length of the path from $n_{i,j}$ to $n_{i,j+\kappa}$ and the path from $n_{q,j}$ to $n_{q,k,j}$ is the same as the length of the path from $n_{q,j}$ to $n_{q,k,j}$. Because of the structure of the grid, $x_{\text{diff}}$ is $2 \times \kappa + 1$ and hence must be odd.

Lemma 4. Let $x_{\text{diff}} = 0$. Given a pair of nodes $n_{i,j}$ and $n_{q,r}$, the contour of $n_{i,j}$ and $n_{q,r}$ is a hexagon, with corner nodes $n_{i,j}$, $n_{i-k,j+k}$, $n_{i-k,j-k}$, $n_{i+k,r-k}$, $n_{i+k,r+k}$, and $n_{i,r}$, if and only if $y_{\text{diff}}$ is odd and $y_{\text{diff}} \geq 3$.

Proof. Similar to the proof of Lemma 3.

Theorem 1. The contour of $n_{i,j}$ and $n_{q,r}$ is a rectangle bounded by the nodes $n_{i,j}$, $n_{i+k,j-\kappa}$, $n_{q-k,r+\kappa}$, and $n_{q,r}$ if and only if $xy_{\text{diff}}$ is even.

Proof. Given that $xy_{\text{diff}}$ is even, either both $x_{\text{diff}}$ and $y_{\text{diff}}$ are even or both $x_{\text{diff}}$ and $y_{\text{diff}}$ are odd. Without loss of generality, assume that $x_{\text{diff}} > y_{\text{diff}}$. Two such nodes are shown in Figure 3.4.

Consider the case when both $x_{\text{diff}}$ and $y_{\text{diff}}$ are odd. Consider the shortest path from node $n_{q-r+j,j}$ to $n_{q,r}$. Clearly, no shortest path between these nodes can pass through the node $n_{q,j}$. In fact, the shortest path is via the node $n_{q-r+j+1,j+1}$, $n_{q-r+j+2,j+2}$ to $n_{q,r}$. Because $y_{\text{diff}}$ is odd, the path from $n_{i,j}$ to $n_{q-r+j,j}$ must be even and the conditions of Lemma 1 apply. Consequently, the contour of $n_{i,j}$ and $n_{q-r+j,j}$ is a rectangle. Since the length of the path from $n_{i,j}$ to $n_{q-r+j,j}$ is $xy_{\text{diff}}$, by definition,
the corner nodes of this rectangle are \( n_{i,j}, n_{i+\kappa,j-\kappa}, n_{i+\kappa,j+\kappa} \) and \( n_{q-r+j,j} \). Hence, nodes \( n_{i,j} \) and \( n_{i+\kappa,j-\kappa} \) are at least two corner nodes of the rectangle.

Next, consider the path \( n_{i,j}, n_{i+1,j+1}, \ldots n_{i+r-j,r} \). Because the nodes can communicate across the diagonal links, this path must be on a shortest path from \( n_{i,j} \) to \( n_{q,r} \). Further, since \( y_{\text{diff}} \) is odd, the path from \( n_{i+r-j,r} \) to \( n_{q,r} \) via the nodes \( n_{i+r-j+1,r}, n_{i+r-j+2,r} \) to \( n_{q-1,r}, n_{q,r} \) must be even. Hence, by Lemma 1, the contour of \( n_{i+r,q+r} \) and \( n_{q,r} \) is a rectangle. Since the length of the path from \( n_{i+r-j,r} \) to \( n_{q,r} \) is by definition \( x_{\text{diff}} \), the remaining two corner nodes of this contour are \( n_{i+r-j+k,r-k} \) and \( n_{i+r-j+k,r+k} \). Since \( x_{\text{diff}} \) and \( y_{\text{diff}} \) to the source and the sink nodes are both 0, they are corner nodes.

The arguments when \( x_{\text{diff}} \) is even and both \( x_{\text{diff}} \) and \( y_{\text{diff}} \) are even are similar to the above arguments. Since \( n_{i+r-j+k,r+k} \) is the same as \( n_{q-r-k} \), when \( x_{\text{diff}} \) is even, the contour of \( n_{i,j} \) and \( n_{q,r} \) is a rectangle that is bound by the four nodes

Figure 3.4: Contour when \( x_{\text{diff}} \) is even
Next, given a contour of $n_{i,j}$ and $n_{q,r}$ with four corner nodes as $n_{i,j}$, $n_{i+\kappa,j-\kappa}$, $n_{q-\kappa,r+\kappa}$ and $n_{q,r}$, it is shown that $xy_{\text{diff}}$ of $n_{i,j}$ and $n_{q,r}$ must be even.

$xy_{\text{diff}}(n_{i,j}, n_{q,r})$ is even only when $x_{\text{diff}}(n_{i,j}, n_{q,r})$ and $y_{\text{diff}}(n_{i,j}, n_{q,r})$ are both even or are both odd. By definition, $\kappa(n_{i,j}, n_{q,r}) = \frac{|q-i|-|r-j|}{2}$, and $\kappa(n_{i,j}, n_{q-r+j,j}) = \frac{|q-r+j|-|j-j|}{2}$. Thus, $\kappa(n_{i,j}, n_{q,r}) = \kappa(n_{i,j}, n_{q-r+j,j})$. $n_{i+\kappa,j-\kappa}$ is a corner node in the contour of $n_{i,j}$ and $n_{q-r+j,j}$. $x_{\text{diff}}(n_{i,j}, n_{q-r+j,j}) = 2 \times \kappa(n_{i,j}, n_{q-r+j,j})$ by the structure of the grid and hence is even by Lemma 1. $xy_{\text{diff}}(n_{i,j}, n_{q-r+j,j}) = 0$, hence, the distance $d(n_{q-r+j,j}, n_{q,j})$ is always equal to $d(n_{q,r}, n_{q,j})$. Therefore, $x_{\text{diff}}(n_{i,j}, n_{q,r})$ and $y_{\text{diff}}(n_{i,j}, n_{q,r})$ are either both even or both odd. Hence, $xy_{\text{diff}}(n_{i,j}, n_{q,r})$ must be even.

**Observation 2.** When $xy_{\text{diff}} = 0$, the four nodes that bound the rectangle in Theorem 1 are $n_{i,j}$, $n_{i+0,j-0}$, $n_{q-0,r+0}$ and $q, r$; i.e., the shortest path between nodes $n_{i,j}$ and $n_{q,r}$ is unique and the contour of $n_{i,j}$ and $n_{q,r}$ contains only this path.

**Theorem 2.** The contour of $n_{i,j}$ and $n_{q,r}$ is a hexagon bounded by the nodes $n_{i,j}$, $n_{i+\kappa,j-\kappa}$, $n_{i+\kappa+1,j-\kappa}$, $n_{q-\kappa,r+\kappa}$, $n_{q-\kappa-1,r+\kappa}$ and $(q, r)$ if and only if $xy_{\text{diff}}$ is odd and $xy_{\text{diff}} \geq 3$.

**Proof.** The proof is very similar to the proof of Theorem 1.

**Observation 3.** When $xy_{\text{diff}} = 1$, $\kappa = 0$ and hence, the six corner nodes that bound the hexagon in Theorem 2 are $n_{i,j}$, $n_{i+0,j-0}$, $n_{i+0+1,j-0}$, $n_{q-0,r+0}$, $n_{q-0-1,r+0}$ and $n_{q,r}$; i.e., the corner nodes of the contour are $n_{i,j}$, $n_{i+1,j}$, $n_{q,r}$ and $n_{q-1,r}$.

**Observation 4.** Every node in the contour of $n_{i,j}$ and $n_{q,r}$ will have one, two or three shortest paths to $n_{q,r}$.

**Proof.** Proof is immediate from the construction of the contour.

The results in this section are used by networked embedded system nodes to compute the contour.
3.2 Uniform spreading

Assume that a node (source) is sending messages to a sink such as a monitoring station. Every message contains the location of the source and the sink$^2$.

Figure 3.5 depicts a scenario when the source is sending 99 messages to the sink. Every node in the contour uniformly spreads all the messages it receives over all the shortest paths from itself to the sink. Because $xy_{diff}$ is even, the contour is a rectangle. The number in each node represents the number of messages that is handled by the node. As may be noticed in the Figure 3.5, the nodes close to the top left diagonal edge of the contour have many more messages than the other nodes.

![Figure 3.5: Uniform Spreading](image)

$^2$The assumption is that every monitoring station that wishes to receive monitoring messages from the networked embedded system will first register itself. This information is percolated through the network and every node is aware that there is a monitoring station that is interested in receiving messages.
Observation 5. A series of simulations using uniform spreading revealed that the nodes along one path in the contour always handled more messages than other paths.

3.3 Orientations of a Contour

Figure 3.6 shows a monitoring station located at the center. The plane is divided into eight octants. For convenience, the octants are named $O_1, O_2, \ldots, O_8$ in the counterclockwise direction as shown.

When $xy_{diff} = 0$, the source node will lie on one of the diagonal lines and the contour comprises a single path (Lemma1). When $x_{diff} = 0$ and $y_{diff} \neq 0$, or $y_{diff} = 0$ and $x_{diff} \neq 0$, the contour will be symmetrical around the non-zero axes. In all other cases, most of the shortest paths in the contour lie in the same octant as the one in which the source node is located.

![Figure 3.6: Orientations of a Contour](image)

Definition 3. When $xy_{diff} \neq 0$, or $x_{diff} \neq 0$, $y_{diff} \neq 0$, the orientation of a contour is the number of the octant in which the source node is located.

The right side of Figure 3.6 shows a contour with Orientation 1. Most of the shortest paths in this contour lie in octant 1.

It was observed through simulations that when the orientation of the contour is 1, nodes on the path closer to the diagonal line always handle more messages than the other nodes in the contour. The nodes that appear in the light colored area in
Figure 3.6 handle most of the messages. Further, irrespective of the orientation of the contour, nodes on the path closest to the diagonal lines, which divide the plane into octants, always handle more messages than other nodes in the contour. Figure 3.7 shows contour in eight possible orientations\(^3\). The solid lines in Figure 3.7 depict the paths on which nodes that handle most messages are located.

![Figure 3.7: Eight possible Orientations for Contours](image)

Let \(\hat{x}_{\text{diff}}, \hat{y}_{\text{diff}}\) and \(\hat{x}y_{\text{diff}}\) represent the values of \(x_{\text{diff}}, y_{\text{diff}}\) and \(x_{\text{diff}}y\), respectively, when the sign is also considered. Table 3.1 shows the conditions necessary for a contour to have one of the eight orientations.

### 3.4 Heuristic Spreading

Based on the above simulation observations, methods to effectively spread the messages over the available paths in a contour were explored. The theorems from section 3.1 were used by each node to compute the contour. The heuristic method for spreading depends on recognizing that each node can also compute the width, \(w\) of the contour at its own location because it knows the distance from itself to the source.

\(\text{Only the light colored part of the contours are shown}\)
and the sink and all the corner nodes of the contour. When the messages are spread ideally, each node will handle $\frac{N}{w}$ messages, where $N$ is the number of messages that must be sent from the source to the sink, as shown in Figure 3.8.

As shown in Figure 3.9, starting from the source, messages must be spread only for the first $\kappa$ steps. After $\kappa$ steps, the nodes select one shortest path towards the sink and not spread the messages as long as the number of nodes at the next step is equal to the number of the nodes in the current step. The nodes within $\kappa$ steps from the sink must aggregate messages from the nodes in the preceding step. The theorems in section 3.1 used by the node to determine the contour.

The three regions of a contour are referred as the Expansion region, Propagation region and the Contraction region and appear in every contour with a small variation between the rectangle shaped contour and the hexagon shaped contour.

**Observation 6.** In the expansion region, when each node is carrying an ideal number of messages, a node in a row of width $w$ does not have sufficient number of messages for two nodes in the next row, i.e., row of width $w + 2$.

**Proof.** Since $\frac{N}{w}$ is the ideal number of messages that must be carried by each node in a row of width $w$, consider $\frac{N}{w} - \frac{N}{w+2} = \frac{2N}{w(w+2)} \leq \frac{2N}{w+2}$ for all $w \geq 1$.

To illustrate the heuristic, consider Figure 3.10. The nodes are in two successive rows of the expansion region in a contour. For convenience assume that the nodes are labeled as shown and that there are odd number of nodes in each row.
Definition 4. Let $w$ be the width of the contour in some row of the expansion region, and $w + 2$ be the width in the next row of the contour. Let $w$ be odd. Let the nodes be labeled with $1, 2, 3, \cdots, \lfloor w/2 \rfloor$ as shown in Figure 3.10. Then, node $i$ sends

$$\frac{i \times N}{w + 2} - \frac{(i - 1) \times N}{w}$$

messages to the node $i$ on the next row. and

$$\frac{i \times N}{w} - \frac{i \times N}{w + 2}$$

messages to the node $i + 1$ in the next row.

For example in Figure 3.10, when $N = 70$, $w = 5$, then, for the diagonal node, node 1 sends 10 messages to node A, node 2 sends 6 messages to node B, etc. and for the node straight below, node 1 sends 4 messages to B, node 2 sends 8 messages to C, etc.

Using this heuristic, node A receives $\frac{N}{w+2}$ messages from node 1; note that node A cannot receive messages from node 2. Node B receives $\frac{N}{w} - \frac{N}{w+2}$ from node 1 and
Expansion

Region

Propagation

Region

Contraction

Region

Figure 3.9: Three Regions of Contour

\[ \frac{2N}{w+2} - \frac{N}{w} \] messages from node 2 and i.e., \( \frac{N}{w+2} \) messages. Similarly all the other nodes to the left of the middle node receive \( \frac{N}{w+2} \) messages. By symmetry, the nodes to the right of the middle node also use a similar spreading rule; except that the node that is the farthest from the middle node must be labeled 1. The middle node must send \( \frac{N}{w+2} \) messages to the middle node on the next row.

Definition 5. Let \( w \) be the width of the contour in some row of the propagation region, node \( i \) sends all its messages to the node \( i \) in the next row.

Figure 3.10: Heuristic Spreading in Expansion
**Definition 6.** Let \( w \) be the width of the contour in some row of the contraction region, and \( w - 2 \) be the width in the next row of the contour. Let \( w \) be odd. Let the nodes be labeled with \( 1, 2, 3, \ldots, \lfloor w/2 \rfloor \) as shown in Figure 3.11. Then, every node \( i \) sends
\[
\frac{i \times N}{w} - \frac{(i - 1) \times N}{w - 2}
\]
messages to the diagonal node on the next row, and
\[
\frac{(i - 1) \times N}{w - 2} - \frac{(i - 1) \times N}{w}
\]
messages to the node straight below.

For example in Figure 3.11, when \( N = 70, w = 7 \), then, for the diagonal node, node 1 sends 10 messages to node A, node 2 sends 6 messages to node B, and node 3 sends 2 messages to node C etc. For the node straight below, node 2 sends 4 messages to node A, and node 3 sends 8 messages to node B etc.

![Figure 3.11: Heuristic Spreading in Contraction Region](image)

When the contour is a hexagon, there are special cases that must be considered. If \( w \) is the maximum width of the expansion region, the width of the propagation region is \( w + 1 \). The expansion heuristic (Definition 4) can be used by labeling the nodes as shown in Figure 3.12. Similarly, when the maximum width of the contraction region is \( w - 1 \) and the contraction heuristic (Definition 6) can be used by labeling the nodes as shown in Figure 3.13.

### 3.5 Summary

In this chapter, first, the contour was defined as the union of all shortest paths and every shortest path differs from other shortest path by at least one edge. The
shape on a contour is either rectangle or hexagon depending on whether $xy_{diff}$ is even or odd, respectively. Despite the availability of multiple paths in a contour, uniform spreading of messages over the paths, results in the nodes in one path being loaded more than the nodes in other paths. The heuristic spreading scheme balances the messages over the paths available in a contour.
CHAPTER IV

SIMULATION RESULTS

Contour guided dissemination was validated via simulation and experiments. Section 4.1 presents the simulation tools used and section 4.2 presents the results of the simulation. The number of messages handled in each node was used as the metric.

4.1 Simulation Approach and Tools

All the code developed for this investigation was developed in the NesC language [2] in the TinyOS environment [2]: targeting the MICA2 platform [1].

4.1.1 TinyOS

TinyOS is a component-based framework in which a set of components are wired together to create applications. Components exchange data and events via bi-directional interfaces. Four components were developed to implement Contour Guided Dissemination. These components are shown in Figure 4.1. DiscreteSensorM was used to generate periodic messages that represent values acquired from sensors. UpdateNeighborListM was used to periodically update the neighbors of each node. DiscreteSensorM sends events to HeuristicDS; this component implement the Heuristic spreading strategy. To implement weighted spreading or uniform spreading, the HeuristicDS components is replaced by a suitable component. TestHeuristicDS is a top-level component that is used to "wire" DiscreteSensorM, UpdateNeighborListM and HeuristicDS into a Contour Guided Dissemination application.
4.1.2 TOSSIM and PowerTOSSIM

The new components were simulated using the TinyOS Simulator, TOSSIM [2]. The messages exchanged between nodes was viewed for debugging purposes using TinyViz [26]- Which is an visualization tool. TOSSIM also includes a default propagation model for the wireless signals between nodes. Concerns about the weakness of the propagation model in TOSSIM are well documented in literature [27]. Since the metric for this investigation was number of messages handled, the known weaknesses were not a hinderance to this investigation.

To ensure that components are valid, they were compiled and executed in 4 MICA2 nodes. To ensure the consistency of the results, a plug-in extension to TOSSIM called PowerTOSSIM was used. PowerTOSSIM is based on an accurate model for energy consumption in the MICA2 platform [1]. When using PowerTOSSIM, it was ensured that each node of each step in the contour used approximately the same amount of energy. Since energy used linearly increase with the number of messages.
handled, this metric was useful to validate results. To ensure that the implementation of Contour Guided Dissemination was free from any other restrictions of the TOSSIM simulator, the components were validated via experiments in a 4 by 4 grid MICA2 nodes.

4.1.3 Data Gathering Tool

Every traces were generated from TOSSIM and the data for analyzing the performance of Contour Guided Dissemination was extracted using a C program developed for this purpose. The analysis tools extracts the number of messages received by a node by matching node ID with IDs in the event trace. The extracted data in output in a manner convenient for plotting the graphs shown in section 4.2.

4.2 Simulation Results

The simulation results reported here were conducted on a grid of $9 \times 9$ nodes. Every node can only communication with 8 of its immediate neighbors (see Figure 3.1). The maximum number of hops to the sink was 8. Two different source nodes were selected to obtain a hexagon contour and a rectangle contour. The selected source sent 100 messages to the sink.

4.2.1 Flooding with Duplicate Elimination

As a baseline measurement, the number of messages received by each node is measured when using flooding. In flooding, the source node sends each message to all its neighbors. Each node that receives a message forwards that message to all its neighbors. This process continues until the message is eventually received by the sink. To reduce the cost of flooding, duplicate elimination was used. Whenever a node received a message that it had already received before, the received message was discarded.

Flooding with Duplicate Elimination (FWDE) is an expensive method for communication. The number of messages received at each node after 30 seconds and after
60 seconds is shown in Figure 4.2. Figure 4.3 shows the messages after 90 seconds and
120 seconds. As may be noticed, all the nodes receive the same number of messages
and if 100 messages are sent, these 100 messages are received at all the nodes. Hence
the energy consumption is really high in the network in order to forward all messages
to the sink.

Figure 4.2: Flooding after 30 (Left) and 60 (Right) Seconds

Figure 4.3: Flooding after 90 (Left) and 120 (Right) Seconds
4.2.2 Gossiping with Duplicate Elimination

Gossiping with Duplicate Elimination (GWDE) is an alternative to flooding that can be used in dense systems. Instead of sending a message to all its neighbors, a node will randomly select one of its neighbors to send the message. Compared to flooding, this approach reduces the total number of messages that are sent. Despite the large number of messages that are sent, because of message loss and the random selection, the end-to-end delivery rate is neither consistent nor reliable. Figure 4.4 and Figure 4.5 shows the progression of messages after 30, 60, 90 and 120 seconds.

4.2.3 Contour Guided Dissemination

Figure 4.6 shows the number of messages in each node for the Contour Guided Dissemination method using uniform spreading. In this case, the contour of the shortest paths was a hexagon and a rectangle. Both these contours are in region 2 and it can be seen that the nodes along the diagonal carry more messages than other nodes in the contour. In other words, it is the heavily loaded boundary in the contour.

Figure 4.7 shows the number of messages in each node for the Contour Guided Dissemination method using weighted spreading. Because the region of the contour is known, the weighted parameters can be chosen so that more messages can be sent away from the diagonal line boundary. Therefore, the heavily loaded boundary condition in uniform spreading can be mitigated. The results are shown for both a hexagon contour and for a rectangle contour.

Figure 4.8 shows the number of messages in each node for the Contour Guided Dissemination method using the heuristic spreading method. It may be noticed that the number of messages in the contour is almost same for all the nodes except the source and the sink.

4.2.4 PowerTOSSIM Result

Figure 4.9 shows the results from PowerTOSSIM simulation. This figure shows the energy consumed in each node on the contour. Comparing Figure 4.9 with Figure
Figure 4.4: Gossiping after 30 (Left) and 60 (Right) Seconds

Figure 4.5: Gossiping after 90 (Left) and 120 (Right) Seconds
Figure 4.6: Contour Guided Dissemination - Uniform Spreading in Hexagon (Top) and Rectangle (Bottom)
Figure 4.7: Contour Guided Dissemination - Weighted Spreading in Hexagon (Top) and Rectangle (Bottom)
Figure 4.8: Contour Guided Dissemination - Heuristic Spreading in Hexagon (Top) and Rectangle (Bottom)
4.8, it can be noticed that the energy consumed in each node is proportional to the number of messages handled by the nodes.

4.2.5 Summary and Discussion

FWDE is the easiest way to disseminate messages in the network, but this approach suffers from tremendous amount of traffic because FWDE neglects the location of the sink and forwards the message to all its immediate neighbors. GWDE has less congestion in the network compared to FWDE. However, the end-to-end delivery rate is not high.

Uniform spreading in shortest-path contour is the simplest form of Contour Guided Dissemination (CGD). Simulation results show that it performs better than simple methods that are based on single path. Due to its simplicity, it has a heavily loaded boundary. Weighted spreading scheme mitigates this problem to an extent. Heuristic spreading performs well. This approach is slightly complex in computation and the routing decision will depend on the source of the packet and the location of the sink and location of the node in the contour. To support the heuristic spreading scheme, the source must include the number of messages it is sending in the packet. This requirement is typically not unreasonable because in many engineering systems, diagnostic messages are sent in response to queries and based on the type of query, it will be possible to estimate the number of messages that must be sent.

Figure 4.10 shows a comparison between FWDE and Contour Guided Dissemination. It was noticed from simulation that since the nodes using FWDE forward every message received, the maximum number of messages in the network grows at $O(n^2)$. On the other hand, it was observed in simulation that the number of messages in the network grows at $O(n)$ when Contour Guided Dissemination was used. The Figure 4.10 illustrates that Contour Guided Dissemination is more scalable than FWDE.
Figure 4.9: PowerTOSSIM - Heuristic Spreading in Hexagon (Top) and Rectangle (Bottom)
Figure 4.10: Scalability Comparison
CHAPTER V

CONCLUSIONS

This thesis presented a new approach, called Contour Guided Dissemination for propagating messages in networked embedded systems that exploits the location of each node and the multiple paths available between a source and a sink.

The union of all of the shortest paths between a source and a sink was defined to be a contour. Results in Chapter III show that when the value $xy_{diff}$ is even, the shape of the contour is a rectangle and when $xy_{diff}$ is odd, the contour is shaped like a hexagon. Because the theorems in Chapter III provide the locations of the corner nodes of the contour, these results are sufficient to allow each node to compute the contour in a distributed manner, and to identify its own position in the contour.

It was shown that when the messages are spread uniformly over the available paths, nodes on one edge of the contour handle more messages than the other nodes in the contour.

A heuristic spreading strategy was presented. It was shown through simulation results that the heuristic spreading strategy ensures that all the nodes in the same row on the contour handle approximately the same number of messages. Simulation results also show that the cost of communication in Contour Guided Dissemination compares very favorably with the cost of flooding. The cost of dissemination in this method increases linearly with the size of the grid.

Although the results were based on a regular topology, these results suggest new methods for dissemination in networks with general topologies. In such networks, some nodes must spread the messages, while other nodes must utilize one of the available paths without spreading the messages further. Identifying the structure
of contours, expansion regions, propagation regions, and contraction regions for networks with general topologies are interesting problems and the method presented here provides a systematic framework in which to carry out future investigations.
BIBLIOGRAPHY


