A NETWORK OPTIMIZATION SOLVER FOR ROUTING IN WIRELESS SENSOR NETWORKS

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Shaminda Asela Samaraweera

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

November 2010

Fargo, North Dakota

North Dakota State University
Graduate School

Title

A NETWORK OPTIMIZATION SOLVER FOR ROUTING

IN WIRELESS SENSOR NETWORKS

By

SHAMINDA SAMARAWEERA

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

ABSTRACT

Samaraweera, Shaminda Asela, M.S., Department of Computer Science, College of Science and Mathematics, North Dakota State University, November 2010. A Network Optimization Solver for Routing in Wireless Sensor Networks. Major Professor: Dr. Kendall Nygard.

Many wireless sensor network applications require energy efficient communication between nodes in the network. Sensor networks are of limited resources. Due to this limitation, the routing between the nodes is one of the important aspects of the life span of the total network. Optimization of the routing algorithm is therefore an important decision point in the design of the sensor network. Our study establishes that optimization can increase the life span of the network. We implement an optimization algorithm in the total network, which is capable of saving energy on communication. The energy saving in communication helped us to increase the life span of the network.

ACKNOWLEDGMENTS

I would like to express my sincere thanks to my advisor, Dr. Kendall Nygard, for his continued support throughout this paper. I am grateful for the ideas, suggestions and the enormous amount of guidance given by Dr. Nygard. Also, special thanks go to my advisory committee members for their input, which helped me complete this paper.

I would also like to thank my parents and my wife for supporting me while I pursued this master's degree.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure | Page

CHAPTER 1. INTRODUCTION

Large-scale wireless sensor networks (WSN) are envisioned to monitor wide environments without network management for long lifetimes. Many applications involve the detection of events which include: battle fields, smart buildings, temperature and Volatile Organic Components (VOC) monitoring in buildings, moisture and fertilizer level sensing in agricultural fields, and the detection of intruders across borders.

In sensor networks, variable numbers of sensors are distributed in a geographical area and can be either static or dynamic. In the scope of this paper, we assume the sensors are static. These static sensors are known as nodes. The capabilities of these nodes typically comprise monitoring the environment, capturing specific information and transmitting the collected data. The transmitted data can be either in raw or preprocessed form. A node can also function to forward the data obtained from neighbor nodes using wireless bearers. A typical network structure is shown in Figure 1.

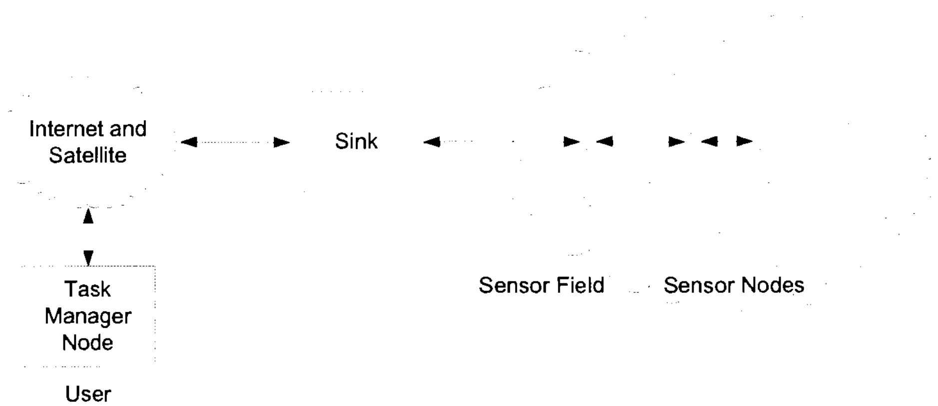

Figure 1. Network set-up of a typical sensor network

The data flow in sensor networks is generally transmitted from the sensor nodes to one or more sensor nodes and finally to the sink. The sink can also communicate with the sensor nodes in a similar manner. Data from the sensor nodes are aggregated at the sink. The sink is similar to a base station with high processing capabilities and may communicate with the task manager node via Internet or satellite.

Sensor nodes are inexpensive and are operated via battery power. The severe resource constraints of WSNs give rise to the need for resource bound solutions. In this paper, we propose a capacitated transshipment problem solving approach to optimize the routing.

1.1. Network Model

The simulation was carried out while considering a heterogeneous sensor network (HSN) environment. The network environment had two types of nodes, low powered and high powered nodes. The low powered nodes, also called L-sensors, used the MICA2 or MICAz motes. The more powerful nodes, H-sensors, used the Imote2 [9]. The H-sensors would form the cluster heads as it has a more computational power and much longer transmission range than the L-sensors present in the network. The environment consists of a small number of H-sensors and a large number of L-sensors due to cost consideration. The H–sensors are used to run the Capacitated Transshipment algorithm on a cluster basis. The results are distributed to the lower powered L-sensors, after deployment of the nodes without the need of deployment knowledge. Furthermore, since H-sensors have a much larger transmission range, they are able to transmit messages directly to L-sensors within their cluster in a single hop. The L-sensors that are not in the transmission range of the H-sensor will communicate with the cluster head through the intermediate L-sensors. The network is assumed to form a maximum of three hops, due to the resource constraint in the low power L-sensors. The limitation of the maximum number of hops prevents energy drains in the L-sensors that facilitate intermediate communication to the cluster head from the L-sensors that are not in the transmission range of the H-sensor. This restriction also helps to prolong the life span of the network.

1.2. Work Overview

Many wireless sensor network applications require energy efficient communication between nodes in the network. Sensor networks are of limited resources. Due to this limitation, the routing between the nodes is one of the important aspects of the life span of the total network. The purpose of this study was to optimize the routing algorithm such that it would help to increase the life span of the network, reduce cost that is associated with redeployment of sensors and minimize the information loss and errors due to sensor failures. The Capacitated transshipment problem has been used to optimize applications such as supply chain, energy distribution, financial transaction etc. In this study we will be implementing the Capacitated Transshipment Algorithm on a Heterogeneous Sensor Network and to evaluate its results. To achieve this we had several objectives and tasks which needed to be completed.

1.2.1. Objective one

The first objective was to programmatically store the network structure, which required several tasks to be completed. The first task was to investigate whether there was an inbuilt tree view class in "Microsoft Visual Studio .net 2008". It did not have an inbuilt tree to store the network data. Implementation of the tree structure can be found in section 2.3.6.

1.2.2. Objective two

The second objective was to implement the Capacitated Transshipment algorithm and its solver on a selected programming language. Several tasks were completed to acquire this objective. The first task was to get an understanding of the algorithm. The next task was to evaluate any improvements that could simplify the implementation. The final task was to see whether the algorithm functioned correctly and gave out the desired results. The implementation of the Capacitate Transshipment algorithm can be found in section 2.2 and 2.3.

1.2.3. Objective three

Our third objective was to run the solver in order to evaluate network connectivity, coverage and cost savings. The first task needed for this was to develop simulation software for the solver. The second task was to experiment using numerous simulations of the schemes under different scenarios. Details regarding the simulation software design and setup can be found in section 2.4. The final task was to compare the simulation results and to evaluate energy savings in the network. This can be found in section 2.4.2.

1.3. Thesis Overview

In this study we implement the Capacitated Transshipment algorithm solver and compare its energy savings with the base case. Chapter 2 discusses the implementation of the solver, the simulation process and the results. Finally we conclude our work in chapter 3.

CHAPTER 2. CAPACITATED TRANSSHIPMENT PROBLEM SOLVER

2.1. Introduction & Existing Work

Network flow optimization models are used in many application areas such as in business, industry, government and the military. Network models with many thousands of nodes and hundreds of thousands of arcs can be routinely solved optimality in a matter of seconds. Specialized data structures for storing and manipulating networks and basis trees play a large role in achieving the tremendous speed at which large scale network optimization problems can be solved. Due to the availability of such tools, network models have been devised for applications far removed from the traditional uses in transportation, and models have become increasingly large and more detailed. In addition, problems with embedded network structure can gain considerable computational advantage by computationally exploiting network sub problems.

In a sensor environment, sense data is sent via a multi-hop wireless communication link to the base station, which is known as the sink. Routing is one of the major areas of concern due to the resource constraints that are present in the Wireless Sensor Network(WSNs). There are many routing protocols proposed in literature that are catered towards WSNs[1-5]. Many optimization techniques have been proposed in the literature [6-8] targeting the WSNs.

Many of the optimization methods use techniques of linear programming. The capacitated transshipment problem is a well known problem in the field of optimization. The early works of Bradley, Brown and Graves [10] and the further work of Brown [11-13] fueled the literature in the capacitated transshipment problem algorithm. Some of the other early adopters of the Capacitated Transshipment problem were Klingman and Glover[14-17]. Many variations from the original problem have been encountered, and solutions have been proposed in literature [18-21]. The transshipment problem has been applied in many application environments [18, 22-24].

2.2. Capacitated Transshipment Problem (CTP)

There are many network flow optimization methods that are proposed in the literature. Many studies use techniques of operational research, heuristics and search techniques to solve the desired problem. The capacitated transshipment problem uses linear algebra to solve the set of equations. The CTP can be modeled as a maximization or minimization problem. In this project, we have modeled and used the CTP to solve a minimization problem.

The basic system equations are as follows. Notation explanation is given in Table 1.

Minimize $z = \sum_{(i,j)\in A} c_{ij} x_{ij}$ (1)

Subject to:

$$\sum_{i:(i,j)\in A} x_{ij} - \sum_{j:(i,j)\in A} x_{ji} = b_i \quad (2)$$

$$0 \leq x_{ij} \leq u_{ij} \quad (3)$$

There are two main constraints that govern the CTP solver as shown by equations 2 and 3. Equation 2 represents a constraint in which given a node, all the flows into the node minus all the flows out of that particular node should be equal to the supply of that given node. The supply b_i for a given node 'i' can be either positive or negative. A positive node

Table 1. Formula notation

A	Directed graph with node set N and arc set A = N x N. A typical element is derived using the following notation. $i \varepsilon N$, $(ij) \varepsilon A$
$\mathbf{c_{ij}}$	Cost or unit of commodity flow on arc 'i' to arc 'j'
$\mathbf{u_{ij}}$	Capacity (upper bound) for commodity flow on arc 'i' to arc 'j'
$\mathbf{b_i}$	Supply of commodity at node i (interpret negative b_i as a demand of $-b_i$ units)
$\mathbf{x_{ij}}$	Commodity flow on arc 'i' to arc 'j'.

b_i represents a supply node, while a negative value for a b_i represents that the respective node is a demand node.

The 3^{rd} constraint that governs the solver, which is shown in equation 3, represents the flow constraint. A flow of an arc i,j represents a range between 0 and the total capacity of that given arc. Due to this constraint in flow, we have a lower bound of 0 and an upper bound equal to the capacity of that given arc.

There are 3 main assumptions that govern the CTP solution.

1) $\Sigma b_i=0$. This is necessary to allow the solver to balance the flow of the total network.
2) The numbers of arcs are more than the number of nodes in the given network. Here the number of nodes represent the number of equations, while the number of arcs represent the number of variables.

   m = number of nodes = number of equations.

   n = number of arcs = number of variables.

   The number of arcs(n) will be much higher than number of nodes(m) in a given network.
3) The capacity constraints will be handled implicitly by upper bounding techniques.

2.3. Methodology

2.3.1. CTP algorithm

The CTP solver was solved using the simplex method and following an iterative process. The overview of the algorithm is listed in Figure 2.

The overview in Figure 2 can be expanded in the following steps.

STEP 1: Initialization

In this step, an initial feasible basis tree is found for the network. One node is designated as the root for the found initial solution.

STEP 2: Calculation of Dual Variables for all the nodes

For each node in the basis solution, dual variable π_i, cost of basis tree path from node i to the root, is calculated in this step. The dual variable is the cost that is incurred when one unit of flow is sent from that particular node 'i' up to the root node.

STEP 3: Pricing

a) For all non basic arcs (i,j), next we compute $r_{ij} = c_{ij} - \pi_i + \pi_j$

Here r_{ij} is the reduced cost of introducing the non basic arc into the current basis solution. c_{ij} is the cost of sending one unit in the non basic arc, while π_i is the dual variable of the tail end of the node of the new arc, and π_j is the dual variable of the head end of the node of the new arc.

b) If $r_{ij} \geq 0$ for all lower-bounded arcs and $r_{ij} \leq 0$ for all upper-bounded arcs, stop, the current solution is optimal.

STEP 4: Mark arc orientation around the cycle

Let arc (k.l) be the most attractive arc to enter the basis, and then identify the cycle

CTP Algorithm

```
function Solver
   InitialBasis()
   bool Optimum = False
   while Optimum= False
      CalculateDual( )
      Compute r_ij for all nonbasic arcs
      If r_ij ≥ 0 for all lower-bounded arcs and r_ij ≤ 0 for all upper-bounded arcs
        Optimum = True
      else
        OrientationMarker()
        Compute θ = min { u_kl ; min {x_ij:(i,j) has (-) sign};min {u_ij – x_ij : (i,j) has (+)
        sign}
        FlowAdjustment()
      endif
   end while
end function
```

Figure 2. CTP algorithm

created when arc (k,l) is added to the basis tree. If $x_{kl} = 0$, mark (k,l) with a (+). If $x_{kl} = u_{kl}$ mark (k,l) with a (-), then give all arcs with the same orientation in the cycle the same mark as (k,l) and give all arcs oriented the opposite way the opposite sign.

STEP 5: Calculating the blocking arc

The blocking arc that is created due to the introduction of the new arc is calculated using the following formula. θ represents the flow adjustment that is necessary in the network due to the introduction of the new arc.

Compute $\theta = \min \{ u_{kl} ; \min \{x_{ij} \};\min \{u_{ij} - x_{ij} \} \}$

STEP 6: Adjust the flows and change the basis.

a) Here we adjust the flows. That is, add θ units to (-)-marked arcs and subtract θ units from (+)-marked arcs.

b) If $\theta = u_{kl}$ go to step 3b.

c) Otherwise, let (p,q) be an arc providing the minimum in θ. Then insert (k,l) in place of (p,q) in the basis tree and go to step 2.

Source: Dr K.Nygards (Fall 2008). The Simplex Method for the Capacitated Transhipment Problem in: CSci 453/653

The above algorithm is executed iteratively until we reach an optimum solution.

2.3.2. Dual variable calculation

After the random deployment of L-sensors and H–sensors, clusters are formed in the network. After all the clusters are formed, we select one cluster. The cluster head is selected as the root of the tree. The nodes are stored in the tree according to connectivity. The next step is to calculate the Dual Variables. The algorithm is described in Figure 3.

One of the important aspects of calculating dual variables is the orientation of the arc. If all the arcs point towards the root, which is shown by the notation (i,P(i)), then λ would be positive and would be equal to the cost of the arc, which is r_{ij} as in line 3. By using the reflection method to make the matrix computation easier, the (P(i),i) arc orientation negates the arc cost as in line 5.

The successor of a given node l in preorder is symbolized by PO(l) and is assigned to a variable A which serves as a depth counter of the tree.

The tree is traversed from top to bottom and left to right, until the dual variables are calculated for all the nodes.

Dual Variable Calculation Algorithm

```
function CalculateDual
    if orientation (i,P(i))
        λ ← r_ij
    else
        λ ← -r_ij
    endif
    Π(l) ← Π(l) + λ
    A ← PO(l)
    Depthl ← D(l)
    While (Depthl < D(A) )
        Π(A) ← Π(A) + λ
        A ← A + 1
    end while
end function
```

Figure 3. Dual variable calculation algorithm

2.3.3. Reduce cost calculation and candidate arc selection

Once the dual variables for each node is calculated the next step is to calculate the reduce cost. The reduced cost r_{ij} of introducing a non basic arc to the solution is calculated using the following formula.

$$r_{ij} = c_{ij} - \pi_i + \pi_j$$

π_i and π_j in the above formula represents the dual variables of the tail end and the head end respectively for the given arc ij.

Once the reduced costs are calculated for all the non basis arcs in the cluster we test whether the current solution is optimum as in the algorithm summary in Figure 2.

If the solution is optimum we stop at this point. If the solution is not optimum we select the non basic arc which provides the reduction as our candidate.

2.3.4. Orientation marker, computing θ and the flow adjustment

Once the candidate non basis arc is selected to we use a back propagation method to find the join. This is illustrated in Figure 4.

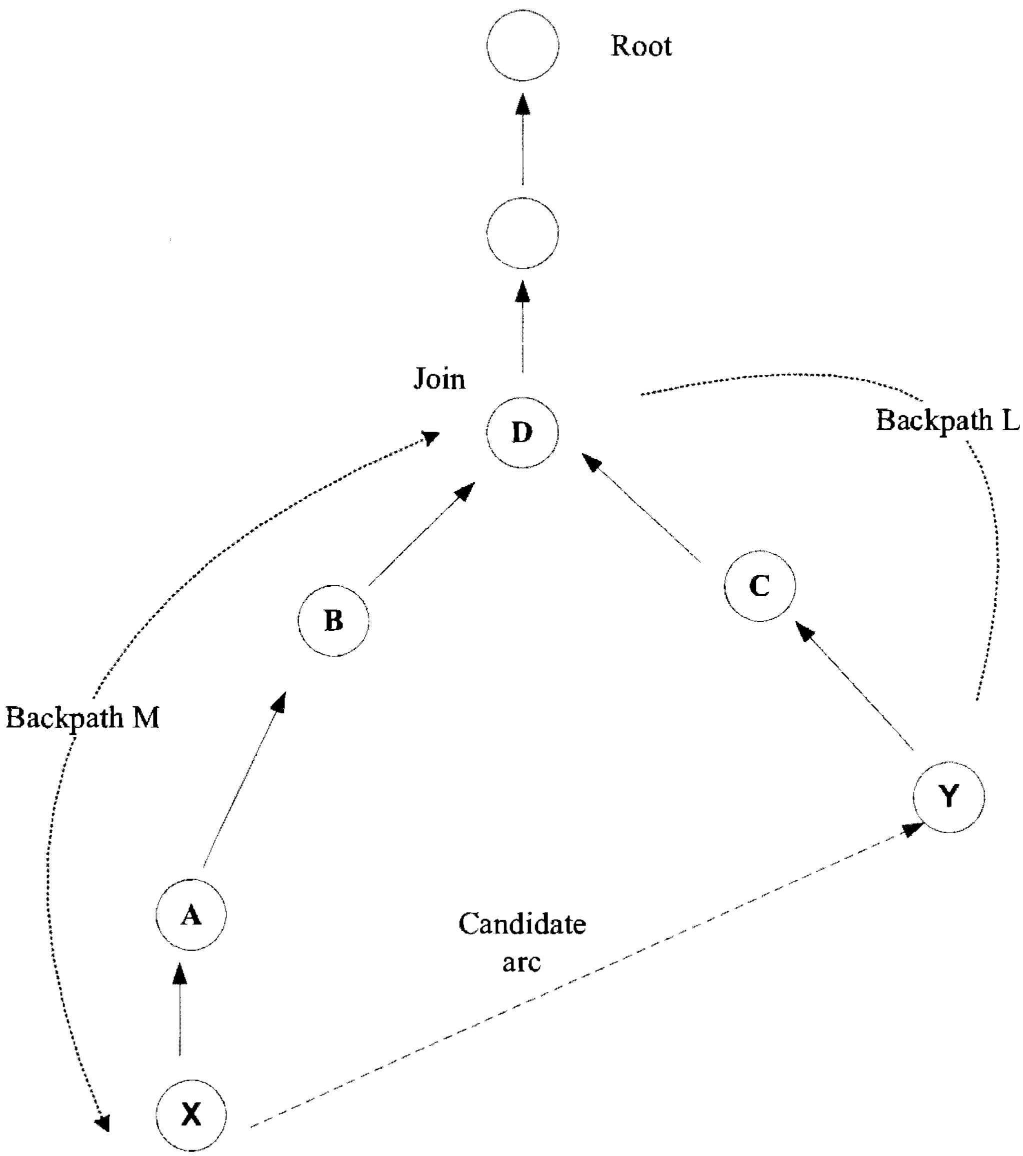

Figure 4. Illustration of backpath propagation

Once the two back paths L and M are found from node X and node Y as in figure X, we calculate the maximum possible units that can be added or subtracted from each path L and M respectively. For the backpath L, which is oriented in the direction of the candidate arc, we calculate the maximum amount of units that can be added without violating the capacity constrain. Once this is found for all the arcs in the orientation of the candidate, we take the minimum value. Next for the backpath M, which is oriented in the opposite direction of the candidate arc, we calculate the maximum amount of units that can be subtracted without violating the constrain of been negative. Here also we take the minimum value. Next we calculate θ which is the minimum of the candidate arcs capacity, minimum of backpath L and minimum of backpath M as follows.

$$\theta = \min \{ u_{kl} ; \min \{x_{ij}\};\min \{u_{ij} - x_{ij} \} \}$$

Once θ is calculated the flows are adjusted accordingly. For all the arcs in the orientation of the candidate arc, θ units will be added to the current flow. For all the arcs in the opposite orientation of the candidate arc θ units will be subtracted from the current flow. The blocking arc, which is the arc which has the minimum value in our θ equation is taken out of the basis while the new candidate arc enters the basis.

If there is a tie in the three parameters used for the θ evaluation, a random selection will be performed to select the blocking arc.

2.3.5. Example solution

The following is a simulated run of the results that were obtained using the implemented solver for a network consisting of one H- sensor and eleven L-sensors.

Figure 5 shows the initial basis solution for a given segment of the network. Node '4_0' is the root node while other nodes are currently at a one hop distance from the root in the initial basis below.

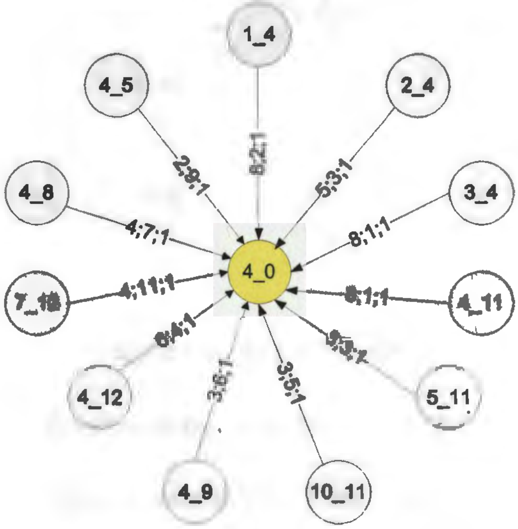

Figure 5. Initial basis solution

For a given arc 'i,j', 'a;b;c' in the above figure represents the arc cost c_{ij}, arc capacity u_{ij} and arc flow x_{ij}, respectively. For example, the arc that flows from node '1_4' to '4_0' has a cost of 8, a maximum capacity of 2 and a current flow of 1 according to the above figure.

Figure 6 represents the optimum solution derived from the solver for the given initial basis solution in Figure 5. The numbers on the arcs represent the arc flows at the optimum case.

Figure 7 represents a listed view that is retrieved from the implemented solver.

Figures 8 and 9 show the simulation extended to the total network. In Figure 8, we need to specify the coordinates of a relevant environment by specifying the top left-hand corner and the bottom right-hand corner. The number of cluster heads and nodes represents

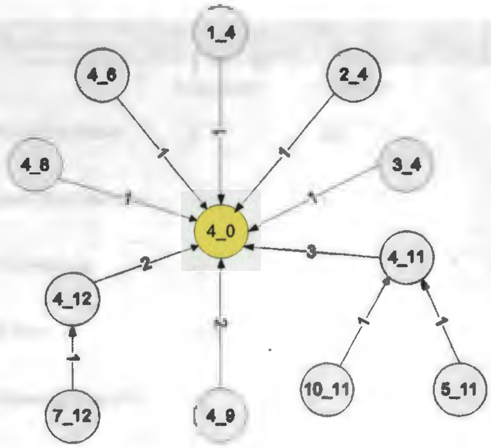

Figure 6. Optimum solution

the cluster heads and nodes in the given environment. The transmission range shows the transmission range of the cluster heads and nodes, respectively.

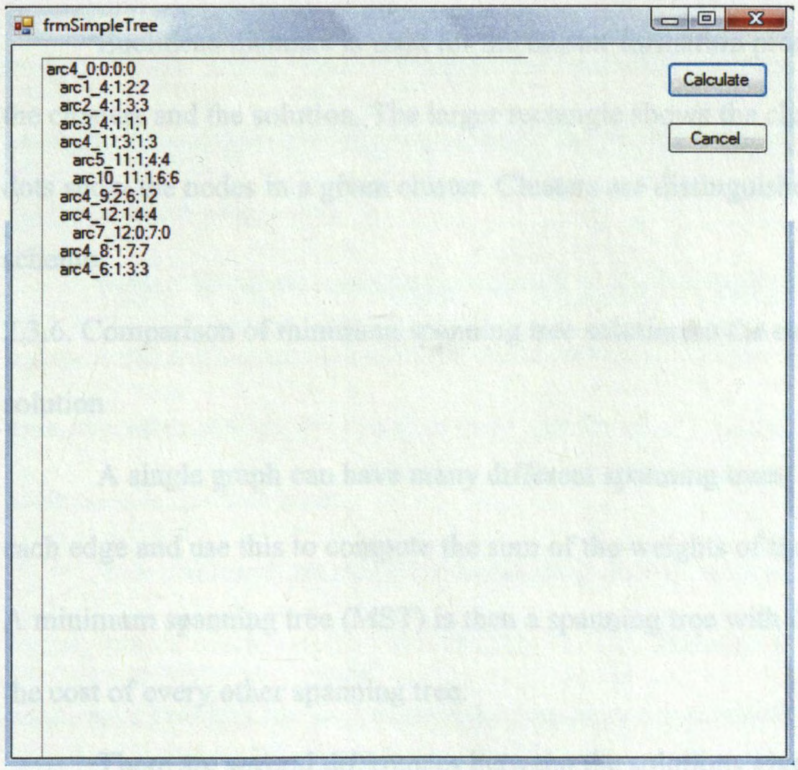

Figure 7. Listed view of the optimum solution

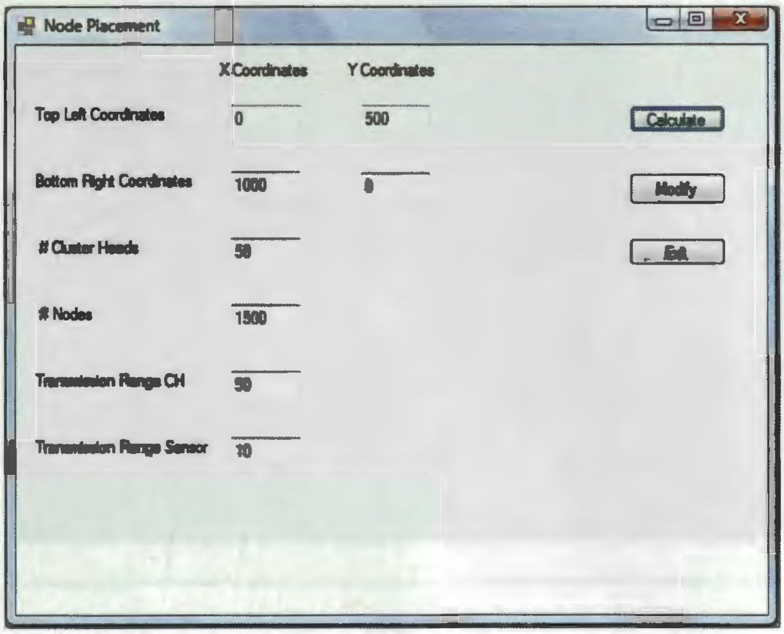

Figure 8. Input parameters for the solver

Euclidean distance is used for the cluster formation process, and Figure 9 shows the clusters and the solution. The larger rectangle shows the cluster heads while the small dots show the nodes in a given cluster. Clusters are distinguished by using a different color scheme.

2.3.6. Comparison of minimum spanning tree solution to the capacitated transshipment solution

A single graph can have many different spanning trees. We can also assign a cost to each edge and use this to compute the sum of the weights of the edges in the spanning tree. A minimum spanning tree (MST) is then a spanning tree with the cost less than or equal to the cost of every other spanning tree.

There are several differences between the solutions given by the minimum spanning tree and the Capacitated Transshipment problem. The major difference is that the spanning

Figure 9. Graphical view of the total network

tree solution is not optimum for the transshipment scenario. Also it treats all nodes in a unique manner and does not distinguish between the root node and the other nodes in the network. Also it does not take into account that the commodity flows towards the root in the transshipment scenario.

Figure 10 shows an example network where we apply the minimum spanning tree solution and the capacitated transshipment solution. The cost of arcs is labeled along the arcs. We assume node 5 is our root node and there will be a unit flow of commodity along each node to the root node.

The minimum spanning tree solution is shown in Figure 11 while the capacitated transshipment solution is shown in Figure 12. The total cost of the minimum spanning tree solution is 11 while the total cost of the capacitated transshipment solution is 10.

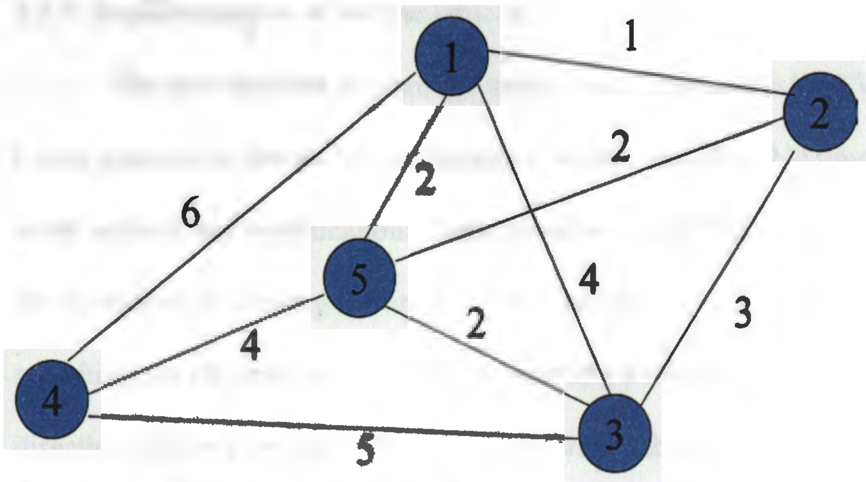

Figure 10. Example network with arc cost

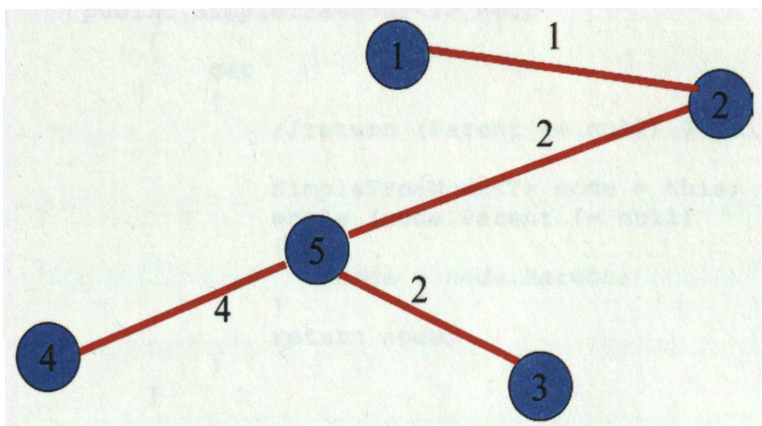

Figure 11. Minimum spanning tree solution

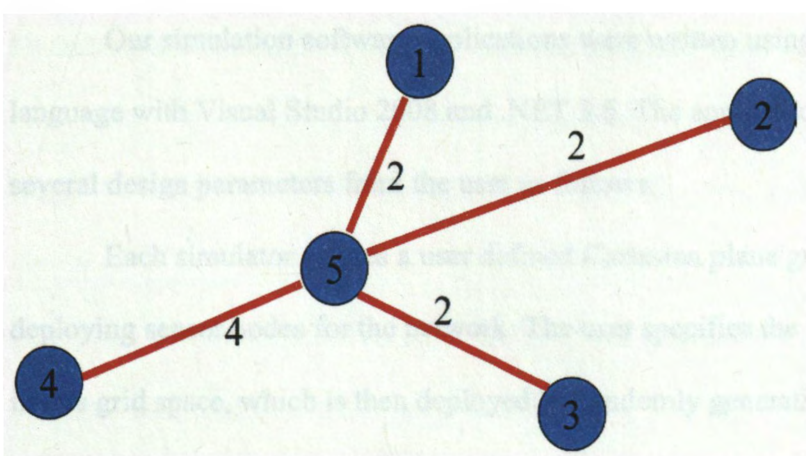

Figure 12. Capacitated transshipment solution

2.3.7. Implementation of the tree structure

The tree structure to store the nodes was developed using "Visual Studio's C#.net". I used generics to design the tree structure as this would enable reuse of the tree in future work without any modifications. Some functionalities that are present in the tree structure we developed are storing the nodes in the tree, searching the parent of a given node, searching the children of a given node, defining a root node, traversing the tree in a given direction, getting the depth of the tree, deriving sub-trees and disposing of the tree.

Example: Code segment for defining the root node

```
public SimpleTreeNode<T> Root
    {
        get
        {
            //return (Parent == null) ? this : Parent.Root;

            SimpleTreeNode<T> node = this;
            while (node.Parent != null)
            {
                node = node.Parent;
            }
            return node;
        }
    }
```

2.4. Simulation Design and Setup

Our simulation software applications were written using the C# programming language with Visual Studio 2008 and .NET 3.5. The application was designed to take in several design parameters from the user as follows.

Each simulator creates a user defined Cartesian plane grid space to be used for deploying sensor nodes for the network. The user specifies the number of nodes to deploy in this grid space, which is then deployed by randomly generating coordinates for each node. This type of deployment method would be similar to an aircraft drop, which does not consider the deployment knowledge that exists prior to distributing sensor nodes into a geographical region. For our simulations, we specified to deploy 100 H-Sensors and 10,000 L-sensors in the region in order to simulate a dense WSN. The simulation was designed such that the transmission range needed to be specified for L-sensors and H-sensors. In our study, we specified our transmission range for L-sensors to be 10 and H-sensors with a transmission range of 50.The above parameters were used as the base case scenario and were changed to obtain various scenarios. The results that were obtained are compared in section 2.4.2.

2.4.1. Simulator process

As shown in Figure 13, the simulator starts after the user inputs are keyed in through a GUI. The user specifies a bounding box of the plain by inserting the top left-hand coordinates and the bottom right-hand coordinates. Next he enters the number of cluster heads and the number of nodes. The transmission range of both the cluster head and the nodes are entered with the iterations that the simulation will be run.

Then the simulator randomly generates the positions of all the H-sensors and all the L-sensors. Once this is completed, the cluster formation process is performed. We assume all nodes have omni-directional antennas and thus, the transmission is also omni-directional. The clustering algorithm is as follows.

Step 1 – For all nodes in the transmission range of a cluster head, assign that particular cluster head.

Step 2 – If there are contentions for cluster heads, as a node can be served by two or more cluster heads, then select the closest one.

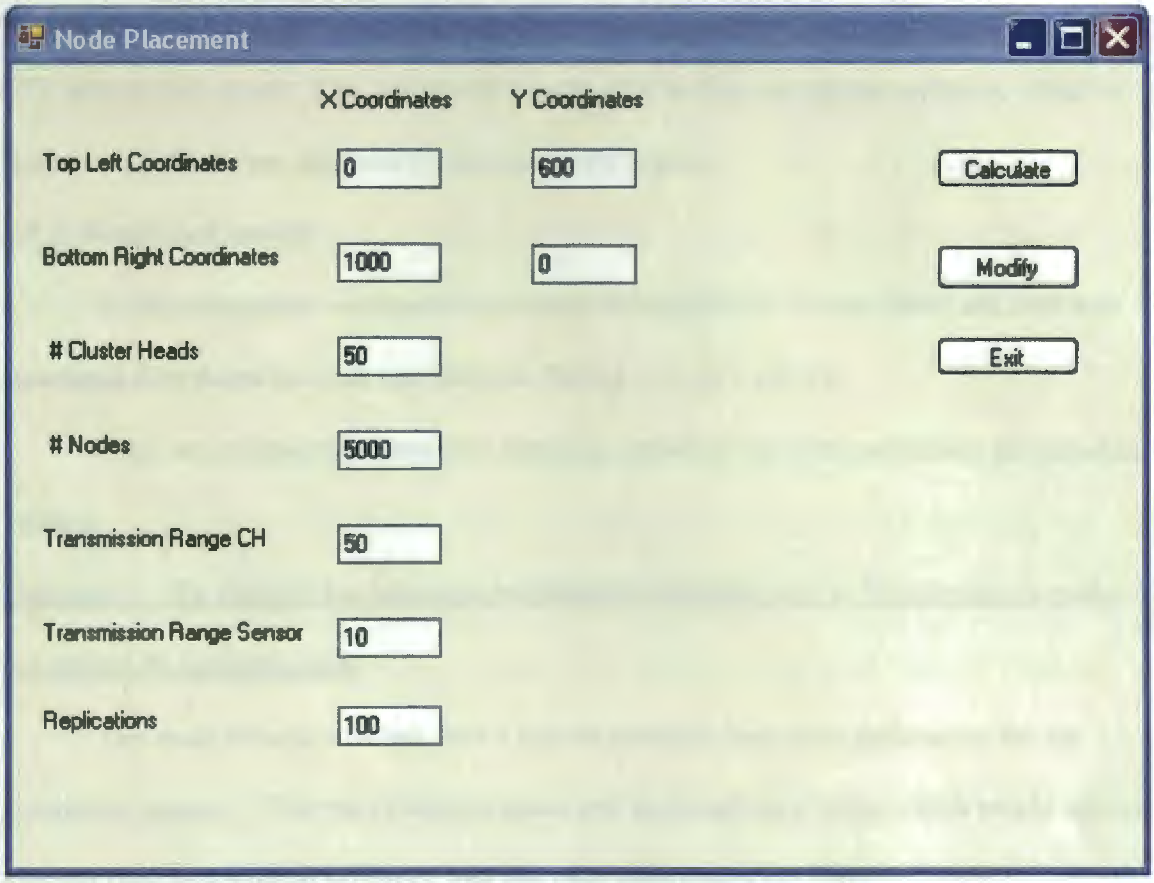

Figure 13. GUI initial parameters

Step 3- If a node is not in the transmission range of a cluster head, then it will try to form a communication link with another node in the first tier.

Step 4 – Once the second tier nodes are established, the same procedure in step 3 will be used to form the third tier. The cluster formation algorithm will end when this step is completed. This is because the communication overhead will be greater for L-sensor in the first and second tier and thus, the life of these nodes will diminish at a faster pace.

Once the clustering is performed on the network, the CTP solver will run on a per cluster basis. The nodes are first stored in the tree structure that was developed by the study. The cluster head will be the root node, and the rest of the nodes will be stored in the trees as derived from the clustering algorithm. Once the nodes are stored in the tree, the CTP solver will initiate. The solver will iterate until it finds an optimal solution, which is stored in a SQL server database for analyzing the results.

2.4.2. Simulation results

In this subsection, we review the results from our simulations. Each test case was simulated fifty times for each test listed in Tables 2, 4, 6, 8 and 10.

First, we evaluate the base case scenario statistics. The test parameters are listed in Table 2.

Test case 1 – To identify the base case parameters with reference to % coverage,% node connected, % cost reduction

The main objective of test case 1 was to establish base case parameters for the simulation process. The rest of the test cases will be based on a range which would include this test case as a median scenario. The test case parameters are listed in Table 2.

Table 2. Test case 1 parameters

Description	**Value**
Number of Cluster heads	100
Number of Nodes	10000
Transmission range of Cluster heads	50
Transmission range of Nodes	10

During the 50 iterations, 82.11% of the nodes were connected in the network. This meant that about 8211 nodes were connected from the 10000 that were randomly deployed in the sensor fields. The 95% confidence intervals were calculated for the base case scenario. The lower limit was 81.06%, while the upper limit of the node connectivity was 83.17%. The connectivity varied with a standard deviation of 3.24%.

During the base case, the coverage had a mean of 86.99% and a standard deviation of 3.33%. This meant that on average 86.99% of the total area was covered by the senors. The lower and upper limits for the 95% confidence intervals were 85.08% and 88.08%, respectively.

When the simulation was run on the base case scenario, it yielded an average cost saving of 7.94% with a standard deviation of 0.44%. This meant that for each unit that traverse the tree there will be on average 7.94% cost reduction compared to the initial basis solution. The lower and upper limits for the 95% confidence intervals were 7.78% and 8.10%, respectively.

Results that were obtained from test case 1 are listed in Table 3.

Table 3. Test case 1 results

Description	**Mean**	**Lower**	**Upper**
Node %	82.11%	81.06%	83.17%
Coverage %	86.99%	85.90%	88.08%
Reduction %	7.94%	7.78%	8.10%

Test case 2 – Changing the number of H-sensors

The focus of test case 2 was to examine the movement of node percentage, coverage percentage and the cost savings with the variations in the number of deployed H-sensors. We varied the number of H-sensors from a lower limit of 20 to an upper limit of 140 with increments of 20. The second test case focused on evaluating sensitivity of change in node connectivity, coverage and cost reduction to the change in the number of H-sensors. The test case parameters are listed in Table 4.

Table 4. Test case 2 parameters

Test #	1	2	3	4	5	6	7
# H-sensors	20	40	60	80	100	120	140

The results for test case 2 from the simulation are listed in Table 5. The increase in the number of H-sensors in the network increased the connectivity, coverage and the cost saving in the network. Connectivity increased from 30.34% to 90.98% when the number of H-sensors increased from 20 to 140. A notable factor here was that the standard deviations also increased at 140 H-sensors, compared to the standard deviation at level 20. Coverage also showed the same trend as the node connectivity and increased from a mean of 35.65% at level 20 up to 93.91% at level 140.

Table 5. Test case 2 results

# H-	**Node %**			**Coverage %**			**Reduction %**		
Sensors	**Mean**	**Lower**	**Upper**	**Mean**	**Lower**	**Upper**	**Mean**	**Lower**	**Upper**
20	30.34	29.86	30.81	35.65	35.10	36.20	1.10	1.08	1.11
40	50.88	50.24	51.52	57.83	57.09	58.57	1.40	1.34	1.46
60	65.32	64.62	66.02	72.17	71.38	72.96	2.79	2.69	2.89
80	75.61	74.90	76.32	81.70	81.00	82.39	4.29	4.11	4.48
100	82.11	81.06	83.17	86.99	85.90	88.08	7.94	7.78	8.10
120	86.90	86.38	87.43	90.80	90.33	91.26	8.36	8.21	8.51
140	90.98	90.47	91.49	93.91	93.48	94.35	8.91	8.64	9.18

A notable factor with reference to node connectivity and coverage was that the tangent at low levels was high compared to the tangent at high levels. Figures 14 and 15 show these results.

The cost saving achieved by running the simulation had the same trend as the connectivity and coverage. Increasing the number of H-sensors improved the connectivity among the L-sensors, thus providing more opportunities for new connections. This enabled

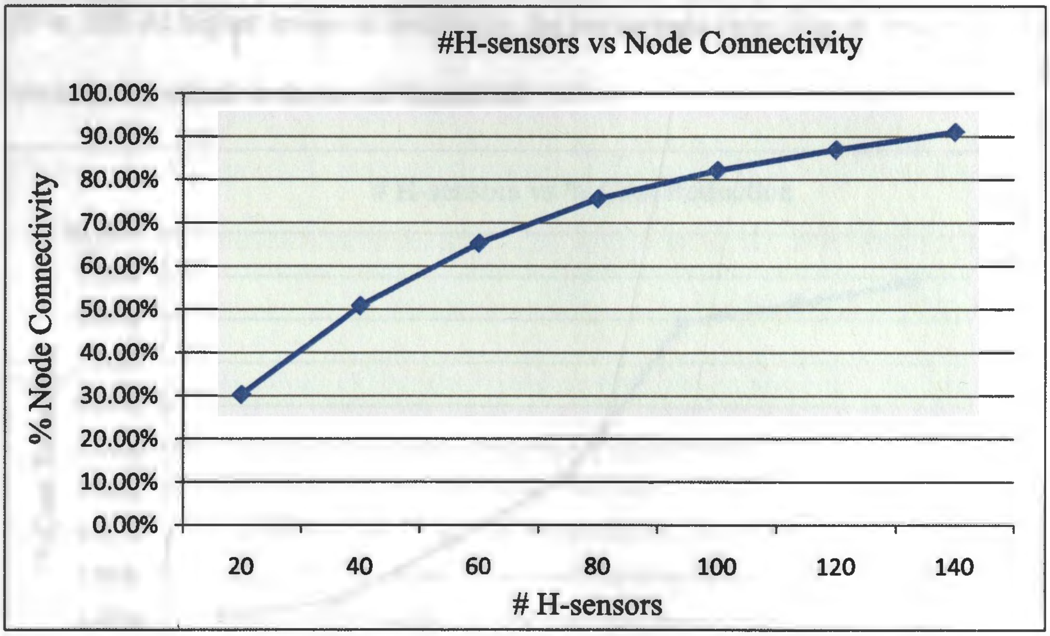

Figure 14. Variation in node connectivity against number of H-sensors

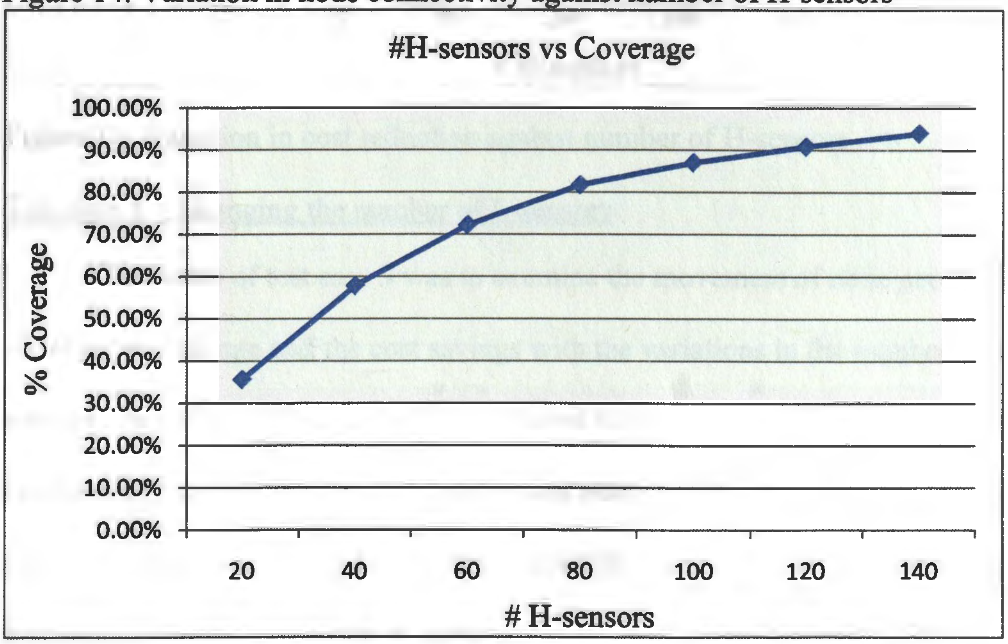

Figure 15. Variation in coverage against number of H-sensors

the increase in reduction in cost at higher levels of H-sensors. A notable factor in this analysis of cost reduction is that the savings almost doubled when the level increased from

80 to 100. At higher levels of H-sensors, the percentage reduction in cost increased at a lower level, which is shown in Figure 16.

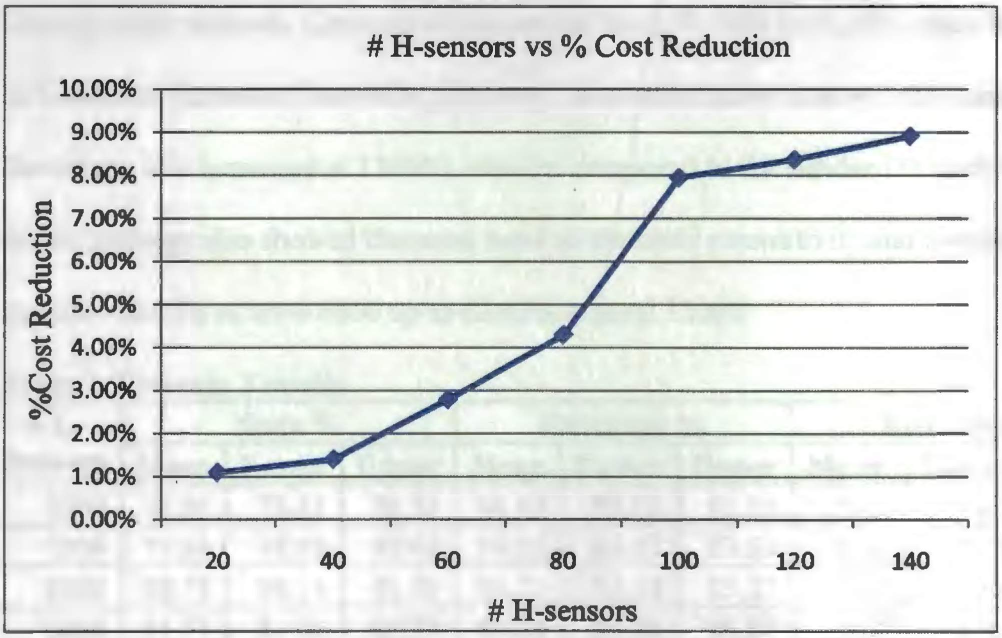

Figure 16. Variation in cost reduction against number of H-sensors

Test case 3 – Changing the number of L-sensors

The focus of test case 3 was to examine the movement of node percentage, coverage percentage and the cost savings with the variations in the number of deployed L-sensors. One of the main questions we wanted to answer was whether there was a relationship between the above 3 outputs and node density. We varied the number of L-sensors from a lower limit of 6000 to an upper limit of 13000 with increments of 1000. The test case parameters are listed in Table 6.

Table 6. Test case 3 parameters

Test #	1	2	3	4	5	6	7	8
# L-sensors	6000	7000	8000	9000	10000	11000	12000	13000

The results for test case 3 from the simulation are listed in Table 7. The increase in the number of L-sensors in the network increased the connectivity, coverage and the cost savings in the network. Connectivity increased from 78.96% to 83.65% when the number of L-sensors increased from 6000 to 13000. A notable factor here was the standard deviations also increased at 13000 L-sensors compared to the standard deviation at level 6000.Coverage also showed the same trend as the node connectivity and increased from a mean of 80.83% at level 6000 up to 89.03% at level 13000.

Table 7. Test case 3 results

# L-Sensors	Node %			Coverage %			Reduction %		
	Mean	**Lower**	**Upper**	**Mean**	**Lower**	**Upper**	**Mean**	**Lower**	**Upper**
6000	78.96	78.21	79.71	80.83	80.13	81.53	5.12	4.67	5.56
7000	79.68	78.93	80.44	82.95	82.27	83.63	5.64	5.29	5.99
8000	80.75	80.14	81.35	84.71	84.14	85.27	5.92	5.60	6.23
9000	81.71	81.06	82.37	86.28	85.70	86.87	7.07	6.93	7.21
10000	82.11	81.06	83.17	86.99	85.90	88.08	7.94	7.78	8.10
11000	82.67	82.03	83.31	87.86	87.24	88.49	8.07	7.61	8.54
12000	83.43	82.75	84.10	88.84	88.26	89.42	8.15	7.98	8.31
13000	83.65	82.93	84.36	89.03	88.38	89.69	9.08	8.83	9.34

Another notable factor with reference to node connectivity and coverage was the tangent at low levels was high compared to the tangent at high levels. The tangent changes were not so significant compared to test case 2 results. Also, doubling the number of L-sensors yielded less than a 5% increase node connectivity and less than a 10% increase in coverage. Figures 17 and 18 show these results.

The cost savings achieved by running the simulation had the same trend as the connectivity and coverage. Increasing the number of L-sensors improved the connectivity, thus providing more opportunities for new connections. There was a significant increase in the connections in tiers 2 and 3 in the network. This enabled the increase in reduction in

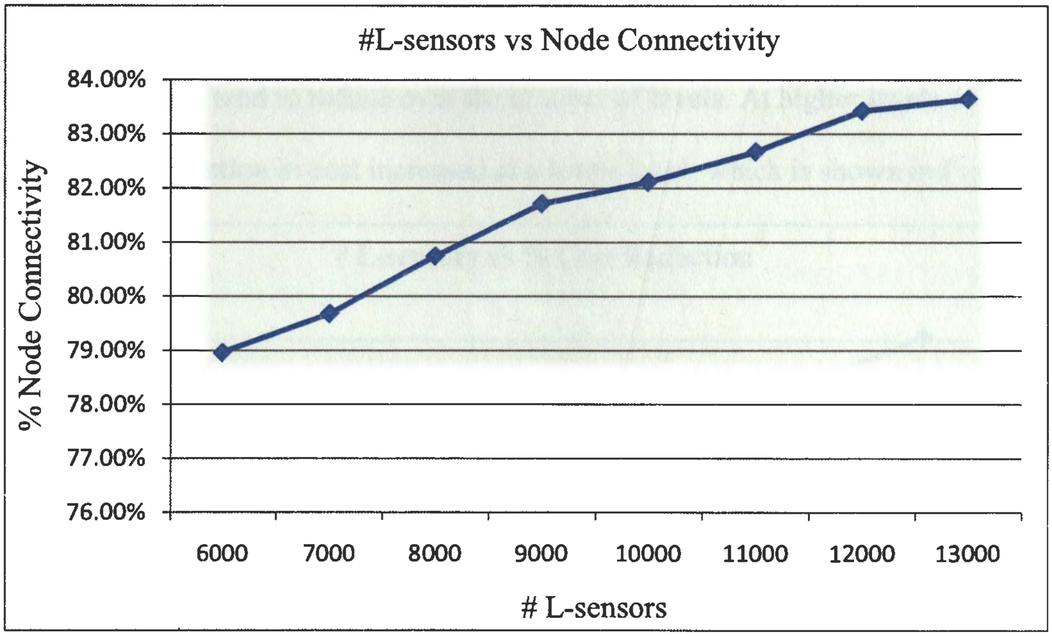

Figure 17. Variation in node connectivity against number of L-sensors

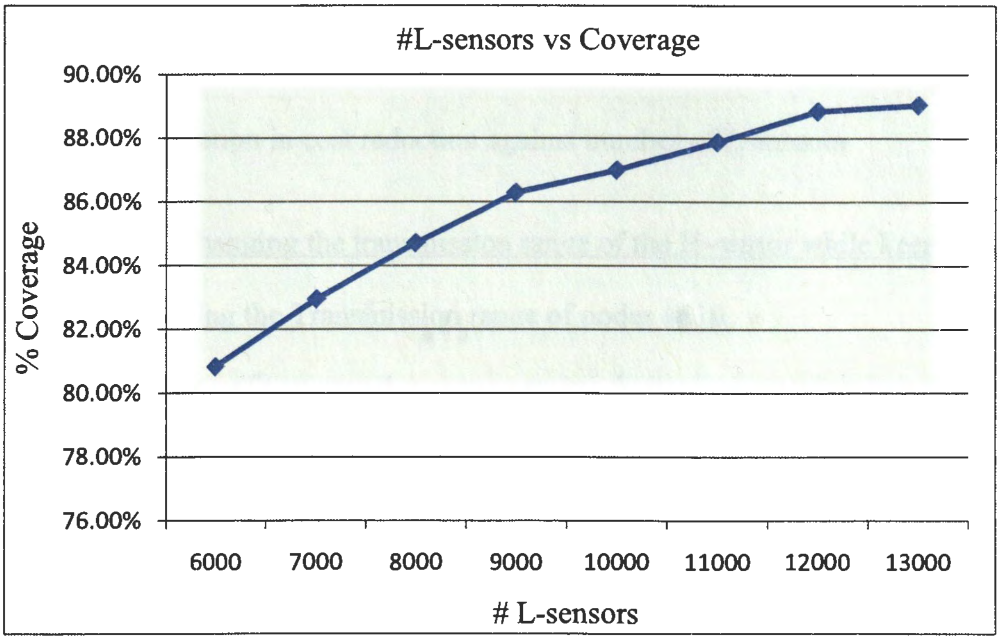

Figure 18. Variation in coverage against number of L-sensors

cost at higher levels of L-sensors. A notable factor in this analysis of cost reduction is that the cost savings tend to reduce over the number of levels. At higher levels of L-sensors, the percentage reduction in cost increased at a lower level, which is shown in Figure 19.

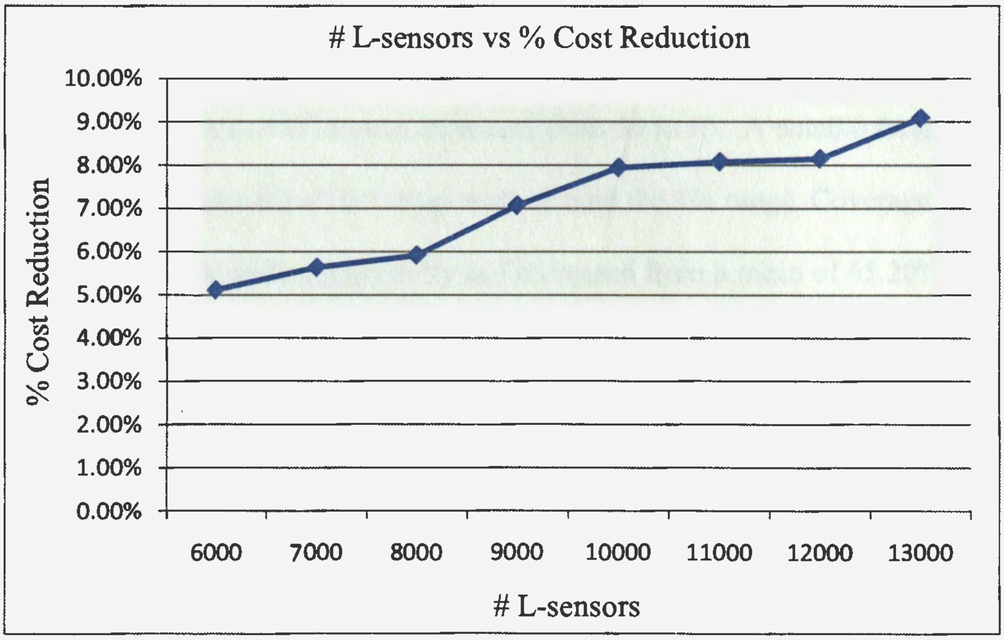

Figure 19. Variation in cost reduction against number of L-sensors

<u>Test case 4 – Changing the transmission range of the H-sensor while keeping all the others constant including the Transmission range of nodes at 10.</u>

The focus of test case 4 was to examine the movement of node percentage, coverage percentage and the cost savings with the variations in the transmission range of the H-sensors. We varied the transmission range of the H-sensors from a lower limit of 30 to an upper limit of 65 with increments of 5. The parameters used in the test case are listed in Table 8.

The results for test case 4 from the simulation are listed in Table 9. The increase in the transmission range of the H-sensors increased the node connectivity, coverage and the

Table 8. Test case 4 parameters

Test #	1	2	3	4	5	6	7	8
Tx H-sensor	30	35	40	45	50	55	60	65

cost saving in the network. Connectivity increased from 55.12% to 92.37% when the transmission range of H-sensors increased from 30 to 65. A notable factor here was the standard deviations for all the cases were around the 1% range. Coverage also showed the same trend as the node connectivity and increased from a mean of 65.20% at level 30 up to 94.27% at level 65.

Table 9. Test case 4 results

Tx H-sensor	**Node %**			**Coverage %**			**Reduction %**		
	Mean	**Lower**	**Upper**	**Mean**	**Lower**	**Upper**	**Mean**	**Lower**	**Upper**
30	55.12	54.55	55.70	65.20	64.55	65.85	2.03	1.86	2.20
35	63.78	63.16	64.40	73.03	72.35	73.71	2.25	2.13	2.36
40	71.25	70.42	72.08	78.90	78.09	79.71	3.66	3.35	3.98
45	77.49	76.80	78.19	83.81	83.12	84.50	4.43	4.19	4.67
50	82.11	81.06	83.17	86.99	85.90	88.08	7.94	7.78	8.10
55	86.74	86.05	87.42	90.50	89.85	91.14	8.29	8.00	8.57
60	89.86	88.96	90.75	92.62	91.81	93.44	9.28	9.03	9.54
65	92.37	91.76	92.99	94.27	93.71	94.83	10.12	10.01	10.23

Another notable factor with reference to node connectivity and coverage was that the tangent at low levels was high compared to the tangent at high levels. The tangent changes were not so significant compared to test case 2 results. Doubling the transmission range of the H-sensors yielded significant increases in node connectivity and coverage. Figures 20 and 21 show these results.

The cost saving achieved by running the simulation had the same trend as the connectivity and coverage. Increasing in the transmission range of H-sensors improved connectivity among the L-sensors, thus providing more opportunities for new connections.

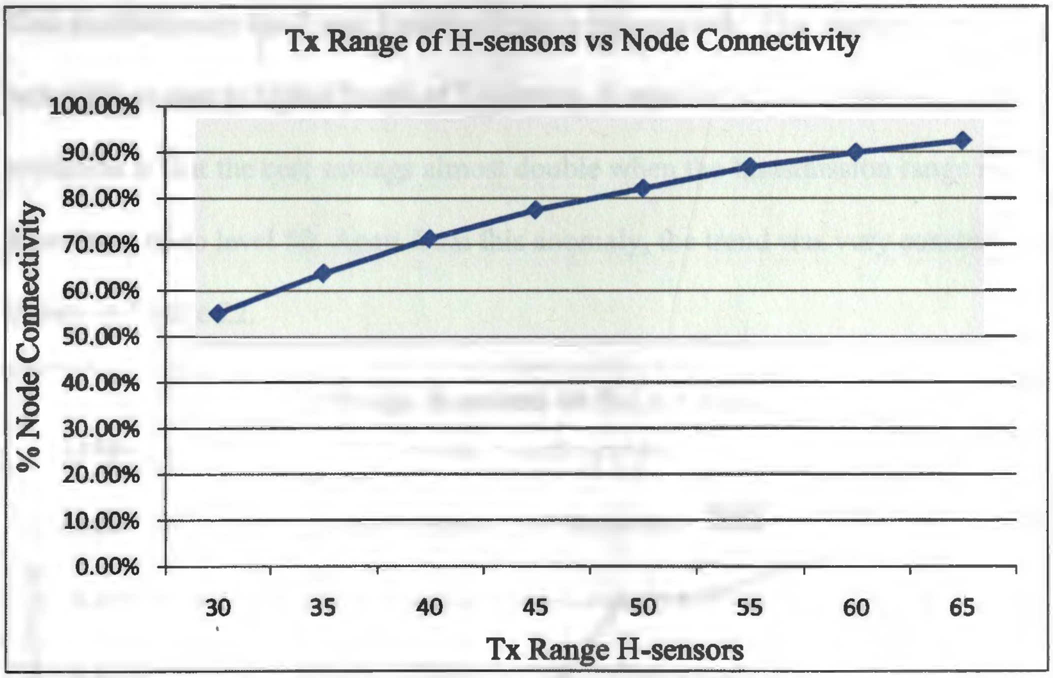

Figure 20. Variation in node connectivity against transmission range of H-sensors

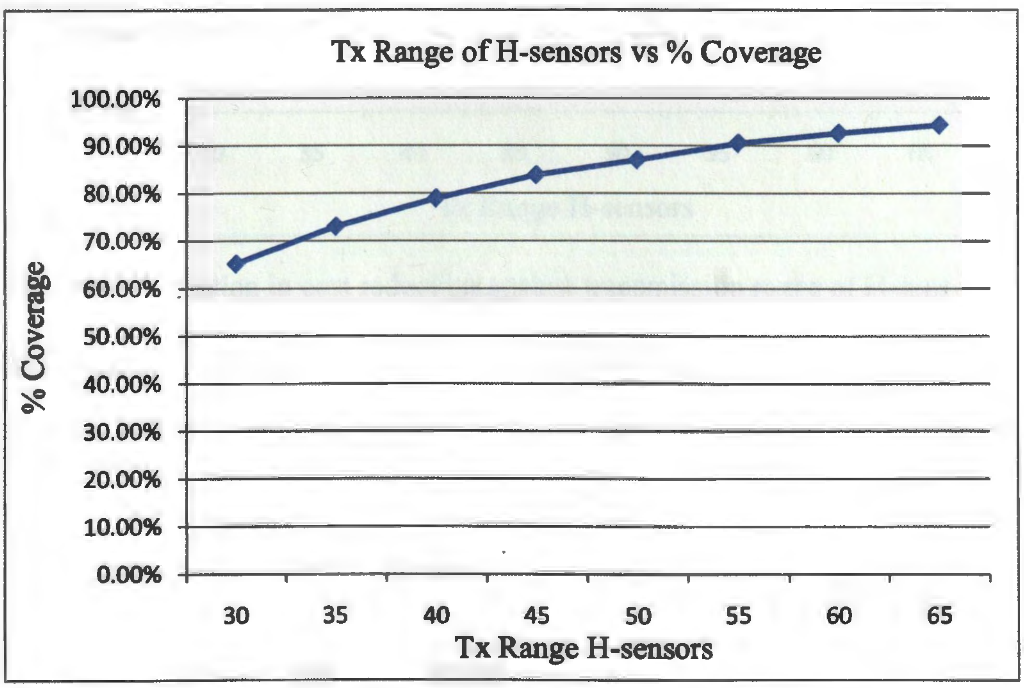

Figure 21. Variation in coverage against transmission range of H-sensors

This enabled more tier 2 and 3 connections in the network. This enabled the increase in reduction in cost at higher levels of L-sensors. A notable factor in this analysis of cost reduction is that the cost savings almost double when the transmission range increased from level 45 to level 50. Apart from this anomaly, the trend was very constant, which is shown in Figure 22.

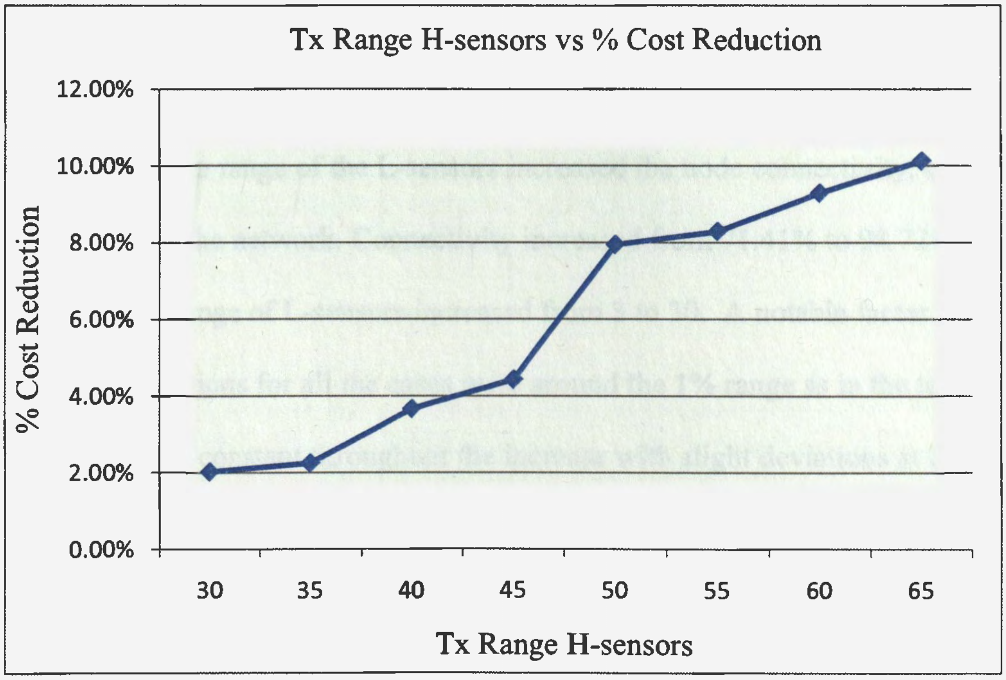

Figure 22. Variation in cost reduction against transmission range of H-sensors

Test case 5 – Changing the transmission range of the L-sensor while keeping all the others constant including the Transmission range of Cluster heads at 50

The objective of test case 5 was to examine the movement of node percentage, coverage percentage and the cost savings with the variations in the transmission range of the L-sensors. We varied the transmission range of the L-sensors from a lower limit of 3 to an upper limit of 30. One of the main questions we wanted to answer was whether the density of coverage had an impact in three outputs of interest. The parameters used in test case 5 are listed in Table 10.

Table 10. Test case 5 parameters

Test #	1	2	3	4	5	6	7	8
Tx L-sensor	3	5	8	10	15	20	25	30

The results for test case 5 from the simulation are listed in table 11. The increase in the transmission range of the L-sensors increased the node connectivity, coverage and the cost saving in the network. Connectivity increased from 71.41% to 98.72% when the transmission range of L-sensors increased from 3 to 30. A notable factor here was the standard deviations for all the cases were around the 1% range as in the test case 4. The trend was very constant throughout the increase with slight deviations at the end points. Coverage also showed the same trend as the node connectivity and increased from a mean of 32.16% at level 3 up to 99.77% at level 30.

Table 11. Test case 5 results

Tx L-sensor	**Node %**			**Coverage %**			**Reduction %**		
	Mean	**Lower**	**Upper**	**Mean**	**Lower**	**Upper**	**Mean**	**Lower**	**Upper**
3	71.41	70.90	71.92	32.16	31.94	32.39	3.81	3.70	3.92
5	71.79	71.20	72.37	58.70	58.21	59.19	3.79	3.69	3.90
8	77.94	77.21	78.66	79.82	79.12	80.52	4.66	4.52	4.79
10	82.88	81.90	83.86	87.89	86.98	88.81	7.94	7.78	8.10
15	91.02	90.38	91.67	95.96	95.52	96.41	10.04	9.72	10.37
20	95.16	94.60	95.73	98.49	98.18	98.80	13.99	13.45	14.53
25	97.63	97.31	97.96	99.41	99.28	99.55	14.84	14.30	15.39
30	98.72	98.40	99.05	99.77	99.69	99.85	16.18	15.61	16.75

Another notable factor with reference to node connectivity and coverage were the tangent at low levels was high compared to the tangent at high levels. Figures 23 and 24 show these results.

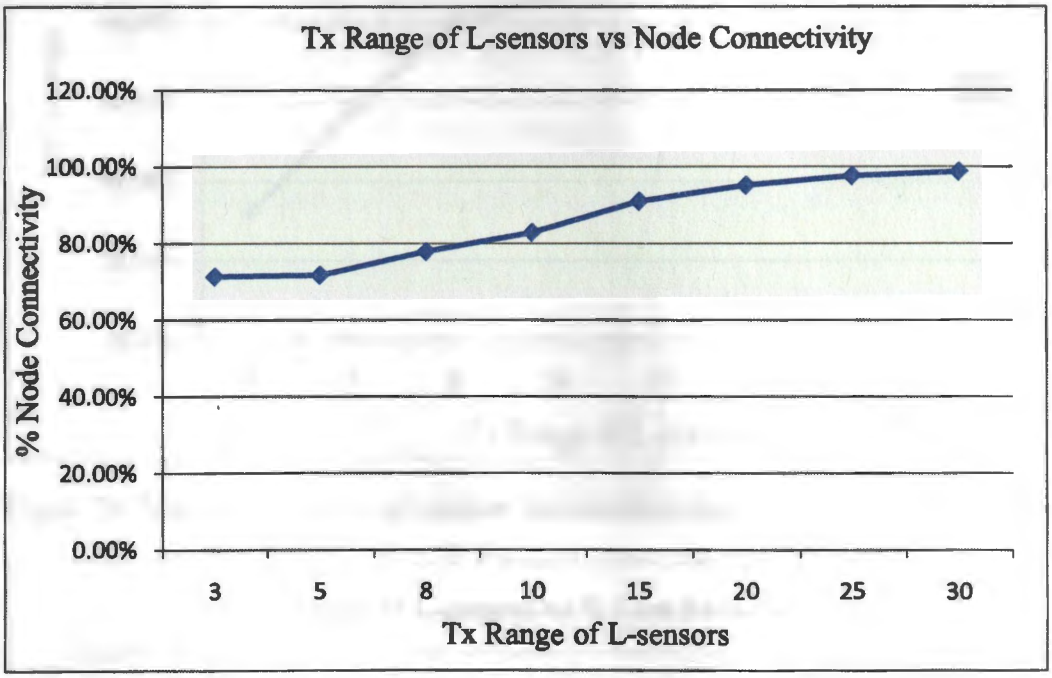

Figure 23. Variation in node connectivity against transmission range of L-sensors

The cost saving achieved by running the simulation had the same trend as the connectivity and coverage. Increasing in the transmission range of L-sensors improved the connectivity among the L-sensors, thus providing more opportunities for new connections. There were significantly more tier 2 and tier 3 connections in the network compared to the other test cases. This enabled to increase the reduction in cost at higher levels of L-sensors. The trend was nearly constant apart from the lower levels, which is shown in Figure 25.

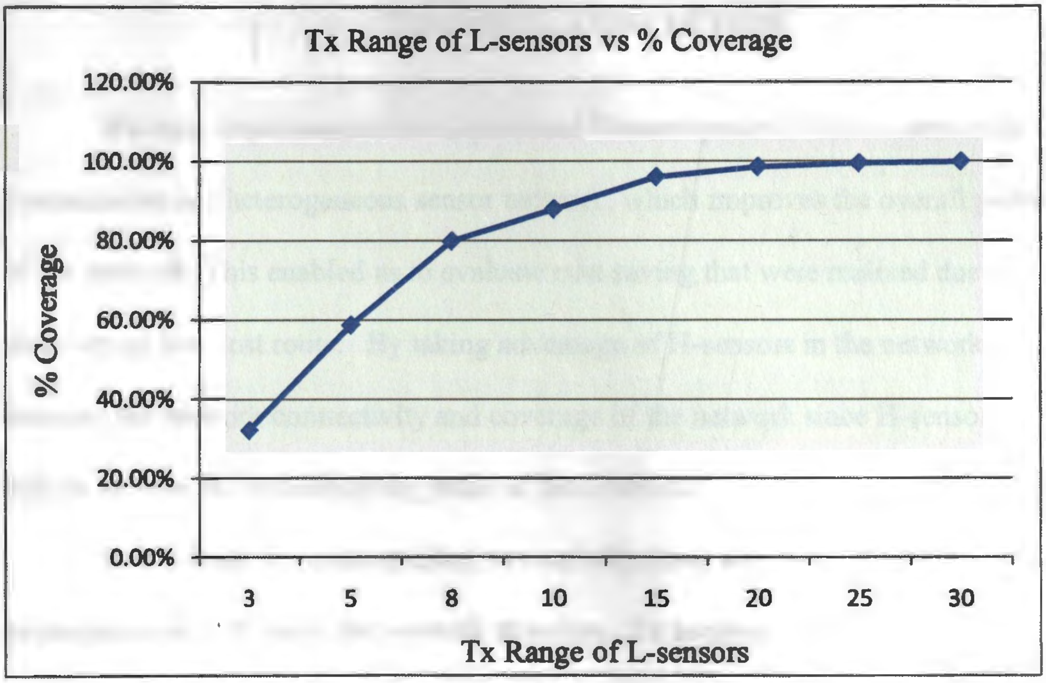

Figure 24. Variation in coverage against transmission range of L-sensors

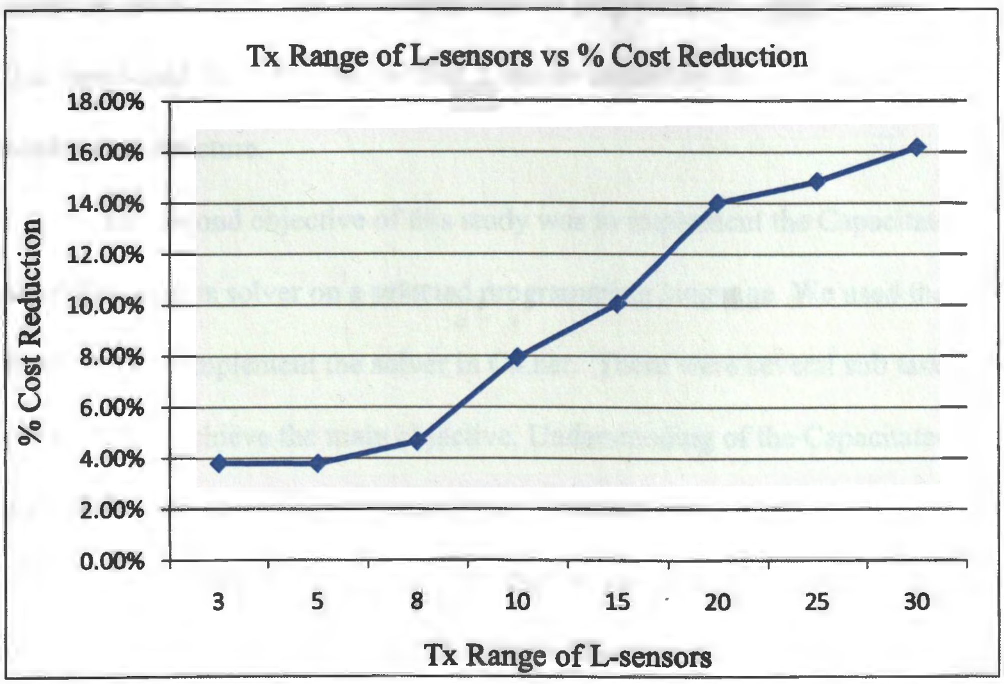

Figure 25. Variation in cost reduction against transmission range of L-sensors

CHAPTER 3. CONCLUSION

We have implemented the Capacitated Transshipment Problem solver for routing optimization in a heterogeneous sensor network, which improves the overall performance of the network. This enabled us to evaluate cost saving that were realized due to discovering low cost routs. By taking advantage of H-sensors in the network, we can increase the network connectivity and coverage of the network since H-sensors should be able to survive the bootstrapping phase of the network.

In this study we accomplished several objectives and tasks. The first objective was to programmatically store the network structure. To accomplish this objective, we have designed a tree structure using generics, which enables us to store the network tree. The usage of generics enabled us to store various properties of nodes and arcs. Also, this code was developed in such a manner that it can be reused by any future project which would need a tree structure.

The second objective of this study was to implement the Capacitated Transshipment algorithm and its solver on a selected programming language. We used the Visual Studio framework to implement the solver in C#.net. There were several sub tasks that were performed to achieve the main objective. Understanding of the Capacitated Transshipment algorithm was performed by using library resources and Dr. Nygard's class notes in CTP. The accuracy of the algorithm was checked by inserting solved optimization problems through the implemented solver. The results obtained from the solver were compared with known results to derive the accuracy of the implementation.

Our third objective of this study was to simulate the solver in order to evaluate network connectivity, coverage and cost savings. We tested the solver using our test cases and showed that node connectivity and coverage improved significantly when the environment was dense. Also, the highest cost savings of 16.75% were achieved when the transmission range of the L-sensors were at the highest. Also, a notable factor was the increase in cost savings when the senor environment was very dense.

REFERENCES

1) Al-Karaki J.N. and Kamal A.E. (2004). Routing techniques in wireless sensor networks: A survey. *IEEE Wireless Communications*.11(6): 6-28.

2) Bruck J., Gao J. and Jiang A. (2009). Localization and routing in sensor networks by local angle information. *ACM international symposium on Mobile ad hoc networking and computing*. 5(1): 1-31.

3) Schurgers C. and Srivastava M.B. (2001) Energy efficient routing in wireless sensor networks. *IEEE Military Communications Conference, MILCOM*.1: 357- 361.

4) Sohrabi K., Gao J., Ailawadhi V.and Pottie G. (2000). Protocols for self-organization of a wireless sensor network. *IEEE Personal Communications*.7(5):16-27.

5) Zhao G., Liu X. and Sun M. (2007). Anchor-based geographic routing for sensor networks using projection distance. *International Symposium of Wireless Pervasive Computing*.

6) Skraba P., Aghajan H. and Bahai A.(2004). Cross-layer optimization for high density sensor networks: Distributed passive routing decisions. *Proceedings of Ad-Hoc, Mobile, and Wireless Networks*.266-279.

7) Wang X., Li Q., Xiong N. and Pan Y.(2008) Ant colony optimization-based location-aware routing for wireless sensor networks. *Lecture Notes in Computer Science*. 5258:109-120.

8) Alippi C. and Vanini G. (2006). Application-based routing optimization in static/semi-static wireless sensor networks. *International Conference on Pervasive Computing and Communications.*

9) (n.d.). Retrieved 06 09, 2010, from Crossbow Technology Inc.[Online]: http://www.xbow.com/

10) Bradley G.H., Brown G.G and Graves G.W. (1977). Design and implementation of large scale primal transshipment algorithms. *Management Science.* 24 (1): 1-34.

11) Brown G.G., Geoffrion A.M. and.Bradley G. H. (1981). Production and sales planning with limited shared tooling at the key operation. *Management Science.*27 (3): 247-259.

12) Brown G.G. and McBride R.D. (1984). Solving generalized networks. *Management Science.*30(12): 1497-1523.

13) Avery W., Brown G.G., Rosenkranz J.A. and Wood R.K. (1992) Optimisation of purchase, storage and transmission contracts for natural Gas Utilities. *Operations Research.*40(3):446-462.

14) Barry R., Elamaz J., Glover F. and Klingman D. (1980). A Network augmenting path basis algorithm for transshipment problems in: Extremal Methods and Systems Analysis (eds: Fiacco A. V. and Kortanek K. O.). Springer-Verlag, Berlin. 250-274.

15) Charnes A., Karney D. Klingman D., Stutz J. and Glover F. (1975). Past, present and future of large scale transshipment computer codes and applications. *Computers & Operations Research.* 2(2): 71-81.

16) Glover F. and Klingman D. (1976). A practitioner's guide to the state of large scale network and network-related problems. *Proceedings of the National Computer Conference*. 945-950.

17) Barr R. Glover F. and Klingman D. (1979). Enhancements of spanning tree labeling procedures for network optimization. *Informs*. 17(1): 16-34.

18) Özdemir D., Yücesan E. and Herer Y.T. (2006). Multi location transshipment problem with capacitated production. *Winter Simulation Conference*. 1470 – 1476.

19) Ekren B.Y. and Heragu S.S. (2008). Simulation based optimization of multi location transshipment problem with capacitated transportation. *Winter Simulation Conference*. 2632-2638.

20) Robinson L.W. (1990). Optimal and approximate policies in multiperiod, multilocation inventory models with transshipmen. *Operations Research*. 38(2): 278 – 295.

21) Tadei R., Perboli G. and Ricciardi N. (2009). The capacitated transshipment location problem under uncertainity. *Cirrelt*.38:1-28.

22) Noche B., Rhoma F.A., Chinakupt T. and Jawale M. (2010). Optimization model for solid waste management system network design case study. *Computer and Automation Engineering*.5:230 – 236.

23) Herer Y.T. and Tzur M. (2001). The dynamic transshipment problem. *Naval Research Logistics*. 48(5): 386–408.

24) Ozdemir D., Yucesan E. and Herer Y.T. (2003). A Monte Carlo simulation approach to the capacitated multilocation transshipment problem. Winter Simulation Conference. 2 :1729 – 1736.