A New Programming Model to Simulate Wireless Sensor Networks : Finding The Best Routing Path

Abrar Alajlan, Khaled Elleithy, and Varun Pande
Department of Computer Science and Engineering, University of Bridgeport, Bridgeport, CT, USA
aalajlan@bridgeport.edu, elleithy@bridgeport.edu , vpande@bridgeport.edu

ABSTRACT

Sensor networks provide a number of extensive programming challenges for Wireless Sensor Networks (WSNs) application programmers. Application developers have proposed various WSNs programming models to avoid these challenges and make WSN programming much easier. In this paper we proposed a new programming model to find the best routing path in WSNs and work on the coding of actual sensor nodes to perform the desired tasks. Then we describe the initial design and the implementation of our proposed model and compare the results when applied in different network topologies with multiple routing algorithms. Finally, we present an evaluation of our model in terms of cost and accuracy.

KEYWORDS

Programming models, WSN, Node- dependent, eventual Consistancy, routing algorithm, cost-minimizing algorithm, heuristic algorithm, routing path.

1. INTRODUCTION

Wireless sensor networks contain a large number of inexpensive sensor nodes deployed to different environments and can be used to detect data and deliver it to sink nodes [1]. Each sensor node is consist of a radio transceiver which has the ability to send or receive packets, a processor to schedule and perform tasks, and power source to provide energy [2]. A sensor network consists of different number of sensor nodes used to transmit and forward sensing data between sensing nodes and the sink or base station. These sensors can be deployed in many applications for example to conduct the environmental conditions such as temperature, humidity and pressures with low power consumption [4]. The ability to place remote sensing nodes without having to run wires and the cost related to it is a huge gain. And as the size of the circuitry of WSN is growing smaller along with the cost, the chances of their field of applications are growing large [3]. Most sensors are battery powered and hence conserving the energy of these sensors is very crucial. Thus, a new model should be applied to manage the power consumption and deliver detected data to the base station.

Several programming approaches have been proposed to assist WSN programming in high-level languages. Two broad classes of WSN programming models have only been introduced recently: local behavior abstraction and global behavior abstraction [5]. The local behavior abstractions can be done at node-level where network programmers need to synchronize the work flow between sensing nodes and maintaining routing code manually. In opposition, global behavior abstractions or equivalently “Microprogramming abstraction”, as we shall discuss later, has emerged as one of the most important aspects in sensor networks. The main objective behind macroprogramming approach is the ability to treat the whole network as one single unit rather than working on each node individually [6].

Since there are many programming models have been explored recently, we would focus on coding the actual sensors to find the shortest routing path, best possible routing path and the maximum routes in multiple wireless sensor networks topologies.
The paper is structured as follows. Section 2 identifies the requirements for sensor network programming. Section 3 provides an overview the related work of macroprogramming model in WSN and section 4 defines the proposed algorithm. Section 5 describes the simulation setup. Section 6 provides the analysis of results. Section 7 discusses the results. Finally, section 8 conclusions.

2. REQUIREMENTS FOR SENSOR NETWORK PROGRAMMING

It is obvious that sensor networks can be used in different applications across multiple environments. Moreover, it is very easy to modify the internal functionality of sensor networks to perform different tasks and to support many sensor networks applications. However, there are four significant requirements for sensor network programming as follow:

- Energy-efficiency
- Scalability
- Localization
- Time-Synchronization

Energy efficiency is one of the most important issues in deigning sensor networks. The overall design of sensor networks should mainly emphasize on enhancing the performance in terms of energy used and power consumed. The total lifetime of a battery-powered sensor networks is limited by the battery's capacity and each sensor node is equipped with a limited computation processor to perform its tasks [7]. Thus, programming model for sensor networks should deploy some applications that attain a proper level of energy-efficiency and to deliver demanded results [8].
Scalability is another important aspect in designing sensor networks applications. A scalable sensor network is a network that able to deliver results with different number of nodes[9]. Since we cannot predetermine the location of sensor nodes and we cannot assure the lifetime of sensor node, programming model should help in such a way to design scalable applications that able to deliver accurate results [9].
In some sensor network applications, nodes are scattered quite randomly in the tested area rather than studying the location of each node. However, the location information of distributed nodes needs to be known in order to exchange data between nodes and the sink. [10]
Localization of sensor network is one of the most important issues in programming sensor networks[11]. Many localization techniques have been proposed recently, either by deploying self-localized technique or by install a Global Positioning System (GPS) device in each node to determine the exact location of sensor node. Moreover, the location of each sensor node can be determined by calculating the distance between the selected sensor node and neighboring nodes [12].
Time-synchronization is another essential requirement for programming sensor network. Clock synchronization is a process used to ensure an accurate scheduling between nodes with no

collision [14]. As stated above WSNs have limited power; therefore, time- synchronization technique reduces power consumption by passing some nodes off from time to time [15].
Clock synchronization in sensing nodes is generally required for some reasons. First, to support the coordination and collaboration between sensor nodes. Second, to manage sleep and active state in each node [16]. Third, to avoid collisions between sensor nodes as used in TDMA [13].
There are some other requirements that we have not introduced in this paper. However, the concentration of this work is mainly on the four requirements stated above.

3. RELATED WORK

Several macro-programming abstractions have been introduced recently. In this section, we provide a brief classification of macroprogramming model as shown in fig.1 and introduce our work under one selected area. Macro-programming model or equivalently "networking abstractions" considered to be high-level WSN programming model where the whole sensor network is treated as a one unit[17]. This approach helps to emphasize on improving the semantics of the network instead of studying the characteristics of the programming environments [18]. Several macro-programming models have been proposed in the past which provide an idea about the node itself and the networking platform. Nano-CF programming framework as in [18] enables to execute multiple applications simultaneously at the same sensor networking platform.
There are two different major classes of network-level abstractions. One is Node-dependent abstractions and the other is Node-independent abstractions. In node-dependent approach a group of nodes can be treated at the same time in one single code. Our proposed work as we will introduce later, falls into this category [19]. This approach designed to allow the user to define the distributed system performance based on the nodes and their states[21].

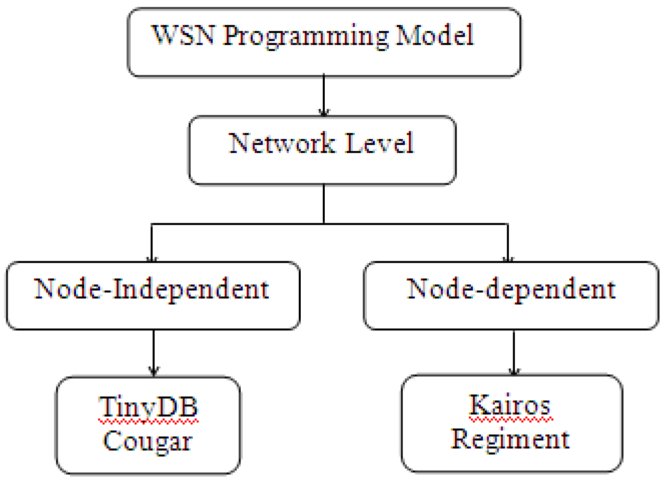


Fig.1 A Classification of Network Programming Models for WSN

3.1. Node Dependent Approach

Kairos is a node-dependent abstraction where the neighboring nodes are able to communicate by using common requests at specific nodes [22].

This approach has a centralized programming environment which can be translated later by the compiler to many executable effective nodal programs [23]. Kairos enhances the use of sensor programming languages by providing three simple mechanisms. First, node abstraction, where the programmer deploys network nodes explicitly and named each node with an integer identifier, yet these integer identifiers do not reflect the structure of sensor node. Hence, there is no need to specify the network structure when using Kairos [23]. Second, identification of one-hop neighbors, where the programmers able to use what called (get neighbors function) to support wireless communication between nodes. When get function called, a list of neighbor nodes returned, so the calling node can select which node to communicate with. Third abstraction is accessing data on a remote node means the capacity to access variables nodes from selected node[20]. Thus, Kairos can be used with many well-known programming languages such as python as in [23].

Another example of node-dependent abstraction is Regiment which is purely microprogramming functional language that allows direct use of program state [24]. However, it uses what called monads; it is described in more detail elsewhere in [25]. In Regiment, programmers deploy groups of data stream or signals. These signals represent the findings of each individual node. Regiment also provides the concept of region as in Abstract Regions [22]. Regions are used to enhance the logical relationship between the nodes and data sharing between sensor nodes. The compiler at Regiment converts the whole program into a form of simple readily program using token machine technique where nodes achieve internal sensing and able to receive signals from neighbor nodes [24].

In addition, Regiment applies multi-stage programming mechanism to support the use of different programming languages that are not maintained by the given program [26]. Also, it approves the use of generics to qualify the program to pass any data types as in C++. It supports three polymorphic data types' stream which represents the changes in nodes state, space which represents the real space with a given value of specific type and event which represents the events that have values and happen at a specific time. The concept behind streams and events is founded in Functional Reactive Programming (FRP); see [27] for more details. Since Regiment is completely functional language, the values of stream, event and space are treated as formal parameters where they can be returned from function and passed as arguments [24].

3.2. Node-Independent Approach

In Contrast, node-independent approach or equivalently "Database approach" is another type of high-level abstractions. This approach used to distributes nodes in a network using independent technique [20]. For Example, TinyDB as in [27] is a query processing system which mainly focuses on improving the energy consumption by controlling tested data. The network treated as one database system where users are able to retrieve information by using SQL-Like queries. This approach should obey what called homogeneous network where all nodes must have same capabilities before testing to achieve the desire result. In TinyDB, data gathered from sensing nodes is actually used as an input in sensor table and system user can access these entries by using SQL-like queries [22].

Cougar is another example of node-independent abstractions. Cougar system is used to test a query processing in sensor networks [29]. Each Cougar system is consist of three level, Queryproxy, Frontend Component and Graphical user Interface. Frist, Query proxy is a tiny database element that runs in sensor nodes to track and perform system queries. Second, Frontend element is used to setup connections between sensor nodes in one network and another nodes in different networks. Third, graphical user interface (GUI) is used to perform queries on sensor networks [29]. Cougar helps to retrieve the data and system behavior. However, it is too difficult to deal with complex applications like tracking system using this technique [28]. Cougar is founded on routing tree which used to attain energy efficiency [22].

Although Node-independent abstraction delivers very simple user interface, it is still not suitable for applications that require a lot of control flow. Thus, the main objective in node-independent approach is to deliver abstraction in order to enhance the sensing type of WSN applications. On the other hand, the main objective for node-dependent approach such as Kairos and regiment is to deliver a wide range of WSN applications that need parallel computations. We summarize the features of each programming abstraction of related work in Table1.

Table 1. Mapping WSN macro-programming abstractions with corresponding features

Key Feature	Kairos	Regiment	Cougar	TinyDB
Node-Dependency	Node-dependent	Node-dependent	Node-independent	Node-independent
Programming Model	Sequential	Functional	SQL-like Queries	SQL-like Queries
Data Access	Sharing the Data	Sharing the Data	Database	Database
Communication Space	Neighborhood based group	Logical group	Whole network	Whole network

4. PROPOSED SOLUTION

The main objective of this work is to introduce a new programming model that is extremely robust to sensor nodes failures.

Although there are several models focus on node-dependent abstractions, we would be working on coding the actual sensor nodes to find the best routing path in different networks topologies. Our proposed work will employ a model of eventual consistency, the sensing nodes are able to deliver the most accurate result even if an internal node is not assured to be reliable.

Therefore, adopting eventual consistency mechanism in our programming model might lead us to new outputs. Figure 2 illustrates the flow chart of our proposed model.

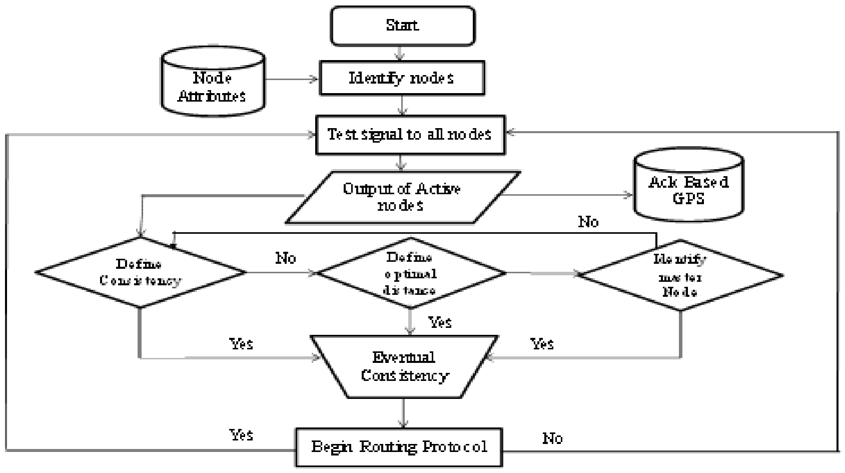


Fig.2 A Flow Chart of Proposed model

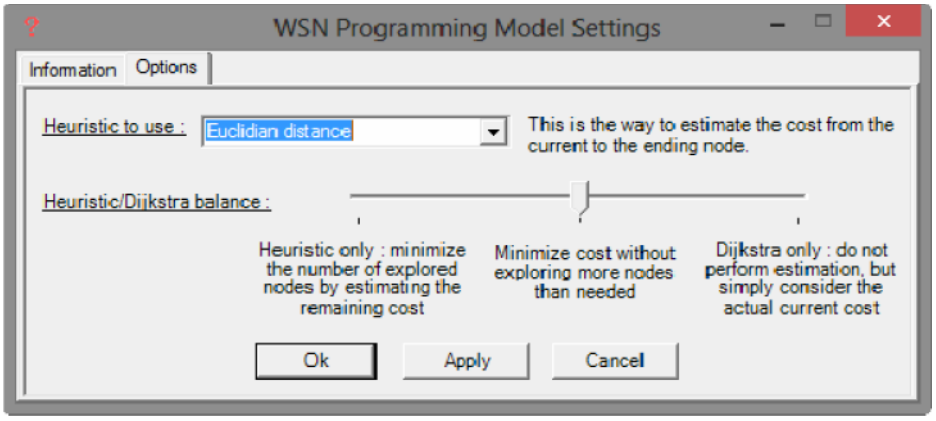
The proposed work is summarized in the following steps:

1. **Identify Node:** Considering the node attributes, a test signal is sent throughout the network to recognize the number of active, sleep and failed nodes. Failure in acknowledgement from any of the nodes leads to the node being understood as compromised or a filed faulty node.
2. **Using tri level logic:** Using a sequence of if-else statements, we built a set of protocols to check the consistency of the nodes by their optimal distance and try to identify their master node. A positive affirmation from any of these logistics leads to the understanding of eventual consistency in between the nodes(we already know the node location and activity from step 2)
3. **Routing protocol:** Considering the active and functional nodes in the network, we can start routing protocol. The routing can be done by the implementation of the basic routing algorithms like the Dijkstras balance and consideration of the Euclidian distance.

The proposed work can be considered as a programming model to find the best routing path in WSNs. The steps above help one determine whether all the nodes in a network are active or functional which allows us to develop a consistent model for the distribution of nodes. The proposed work we have shown previously can be applied in various node architectures. Some of which we have shown in the simulation results later in this paper.

5.SIMULATION SETUP

In this simulation, we are going to explain how the proposed programming model helps us to find the best routing path from selected source to a selected destanition. Consider that we have different number of nodes, which are placed randomely in a network. If one node failed, the system still able to deliver the most accourate result. In this programming model, we can define the number of input and output for each node and the location of source and the destination nodes. Initially all nodes are in a sleep state and placed randomely on a screen. Then, we send a test signal to all nodes to make sure that all the nodes are working. Once the nodes are distributed, the option window allows to choose the various default set values or change the routing algorithms based on the need. It allows one to choose the actual coding of network to find the shortest path, the best possible routing path or find the maxiumu routes by running them on the same network topology, as shown in Figure 3 and Figure 4. After all we are working on a new programming model where the simulation allows choosing the best routing model based on the consistency of the nodes distrubutions.



WSN Programming Model Settings

Information | Options

Heuristic to use : Euclidian distance

This is the way to estimate the cost from the current to the ending node.

Heuristic/Dijkstra balance :

Heuristic only : minimize the number of explored nodes by estimating the remaining cost

Minimize cost without exploring more nodes than needed

Dijkstra only : do not perform estimation, but simply consider the actual current cost

Figure 3: Options window to choose default set values for the proposed model

In Figure 4, all nodes are distributed randomly in a network in order to find the best possible routing path. Though this is based on one of the routing algorithms from option menu. Almost every WSNs is immune to have a few failed nodes or compromised nodes. Keeping such inconsistencies in mind we try to develop a new model that enable users to select the best routing path in any network just by considering node attributes and the consistency characteristics of sensor networks.

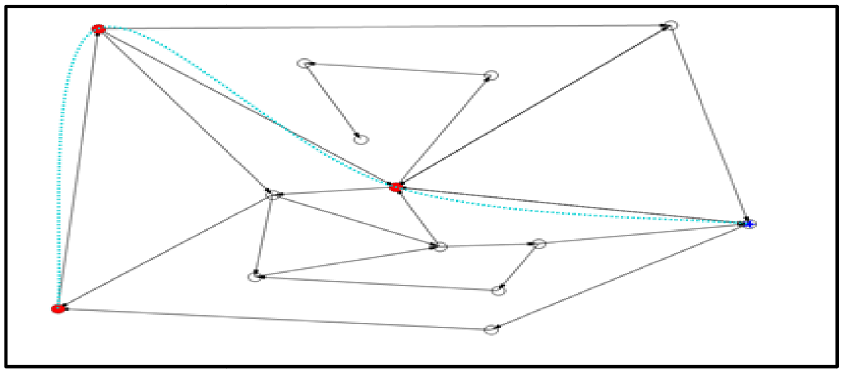


Figure 4: The simulation of the proposed model shown the best possible path from source node to destination node

6. Analysis of Results

We have applied the our model that proposed in the preceding section, and we have tested it to find the best possible routing path. As stated earlier, our model let the user to select the routing algorithm to find the shortest path, best possible path in the same network topology. In this section, we sketch the details of our programming model by examining it in three different well-known network topologies star network, fully connected network and mesh network.

6.1 Heuristic only Algorithm

Heuristic algorithm is used to find the routing path fast and easily with no promise that the best routing will be found. It is simply minimize the number of explored nodes by estimating the remaining cost.

Figure 5, 6, and 7 show how the algorithm work with different network topology, star, fully-connected and mesh networks respectively. The left side shows the mechanism of finding the routing path in some sequential steps. The right side shows the routing path from source node to destination node.

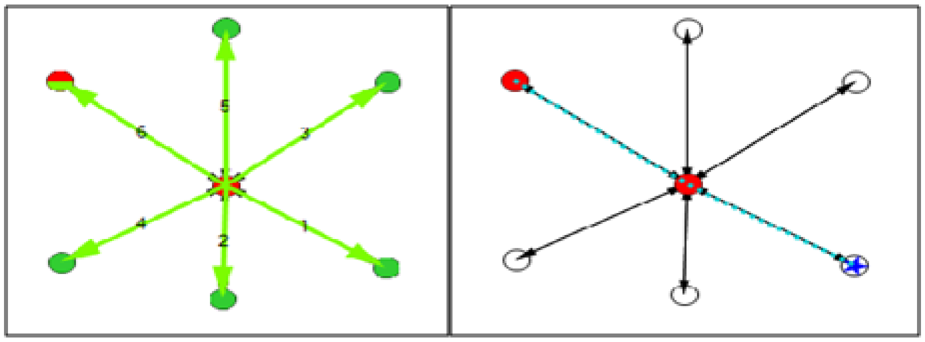


Figure 5: Applying heuristic algorithm to star network

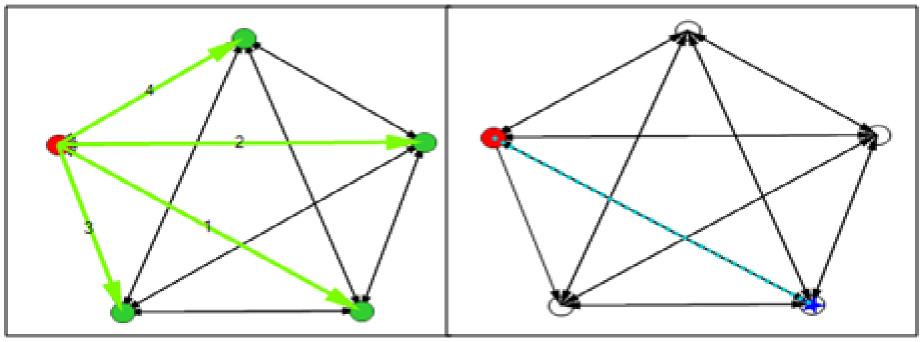


Figure 6: Applying heuristic algorithm to fully-connected network

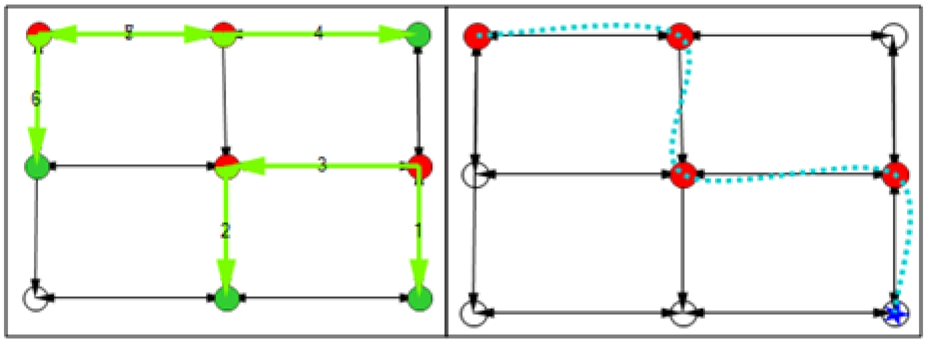


Figure 7: Applying heuristic algorithm to mesh network

As seen on the above graphs, the routing starts with exploring the cost of the nearest node and estimates the cost for other nodes. Heuristic algorithm is used to find the routing path close to the best one fast and easily. In some cases, the estimated cost is equal to the actual cost as shown on table2. However, we cannot guarantee that this routing algorithm deliver the best routing path until it is proven.

6.2 Costs- Minimization Algorithm

Cost minimizing algorithm is used to minimize cost without exploring more nodes. It is simply find the best possible routing path in shortest time.
Figure8,9, and 10 show the cost minimizing algorithm with star, fully-connected and mesh networks respectively. The left side of graph, shows how the model able to find the shortest routing path without consuming too much energy. The right side shows the routing path from source to destination.

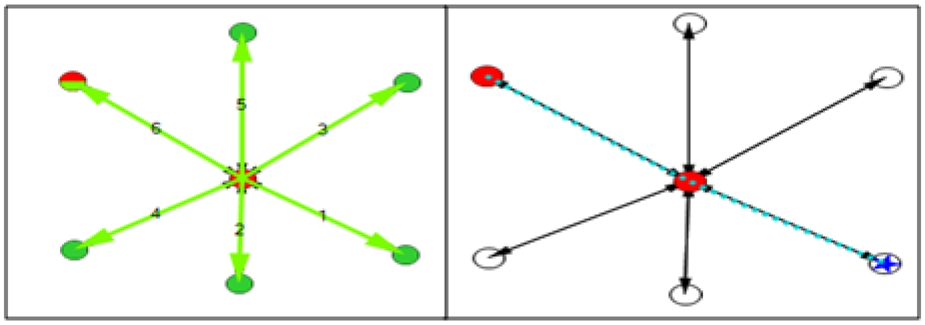


Figure 8: Applying cost-minimizing algorithm to a star network

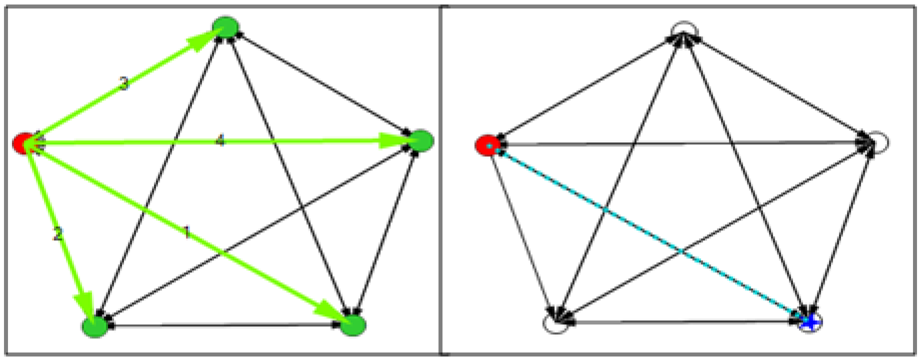


Figure 9: Applying cost-minimizing algorithm to fully-connected network

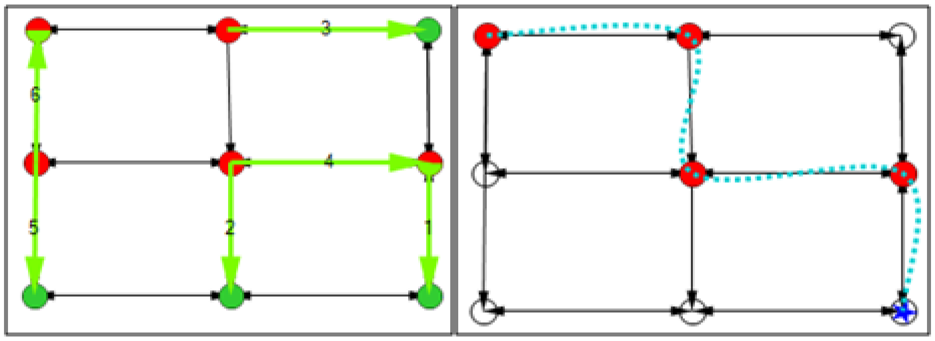


Figure10: Applying cost-minimizing algorithm to mesh network

From above graphs, cost-minimizing algorithm is used to reduce the average cost required to find the best routing path. Basically, it is aim to reduce the number of routing steps in order to find the best possible path.

6.3 Dijkstra only Algorithm

Dijkstra’s algorithm used to find the cheapest routing path between two nodes. It simply does not perform cost estimation, but basically consider the actual current cost of the path from source node to destination node.
Figures 11,12, and 13 show dijkstra’s algorithm with star, fully-connected and mesh networks respectively. The left side of the graph calculated the number of steps to find the cheapest path from source to destination nodes, and the right side show the final routing path.

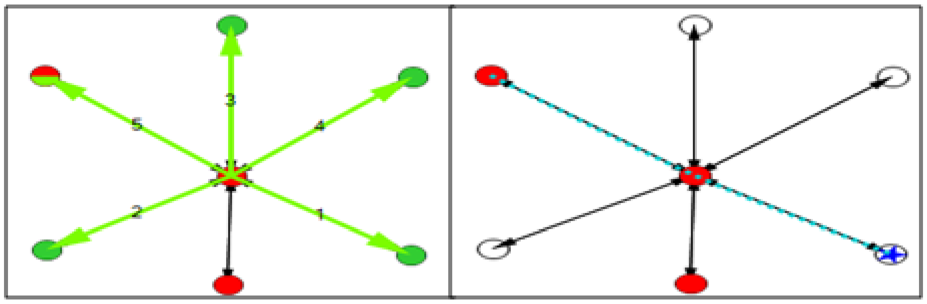


Figure11: Applying Dijkstra’s algorithm to star network

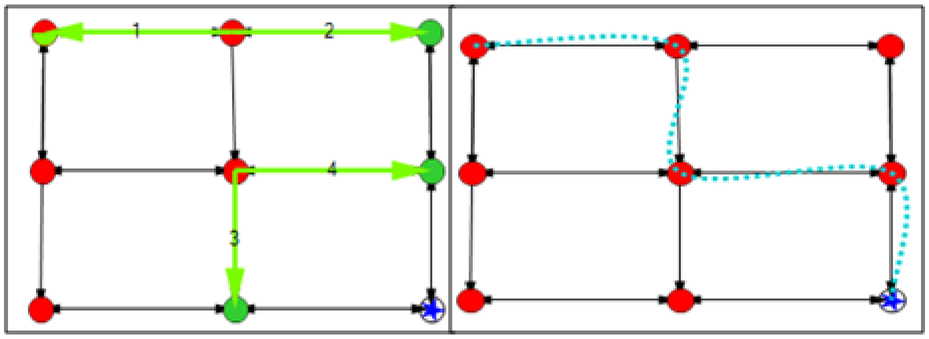


Figure12: Applying Dijkstra's algorithm to fully-connected network

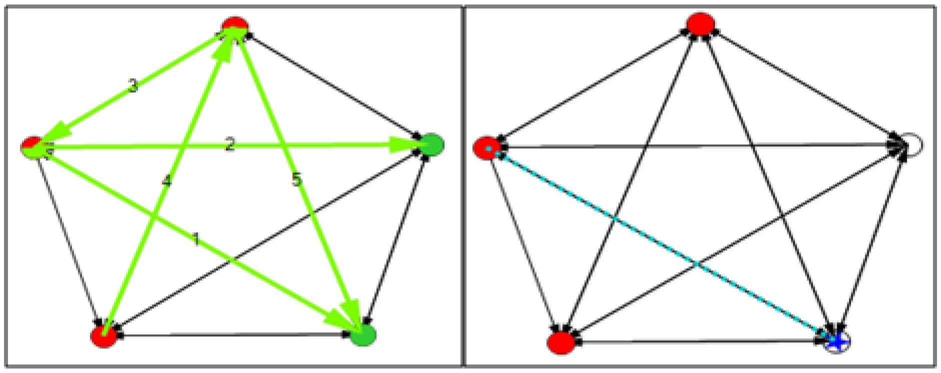


Figure13: Applying Dijkstra's algorithm to mesh network

Dijkstra's algorithm starts from source node as illustrated on above graphs and explore another path in each repetition. Then the process will be repeated until it discovers the whole network and calculate the sum of their costs in order to obtain the actual cost of the routing path.

7. Discussion of Results and Evaluation

The main objective of this paper is to provide a new programming model in WSNs to find the best routing path. We adopt a model of eventual consistency; the system is able to deliver the most accurate result even if an internal node is not assured to be reliable. Moreover, we consider three different routing algorithms, first is heuristic algorithm which is simply minimize the number of explored nodes by estimating the remaining cost. Second, is cost- minimizing Algorithm which is basically minimizing the average time needed to discover the best possible routing path without exploring more nodes. Third is Dijkstra's algorithm which is used to find the actual cost from source to destination nodes.

We tested our model on three different topology star, fully-connected and mesh networks for better understanding. The proposed model can be applied in different networks topologies such as random network as in Figure 4. However, since the random network has a random number of nodes we cannot compare it with any other well-known topologies. Figure 14 and Table 2 show a comparison between heuristic, cost-minimizing and Dijkstra algorithm in term of cost.

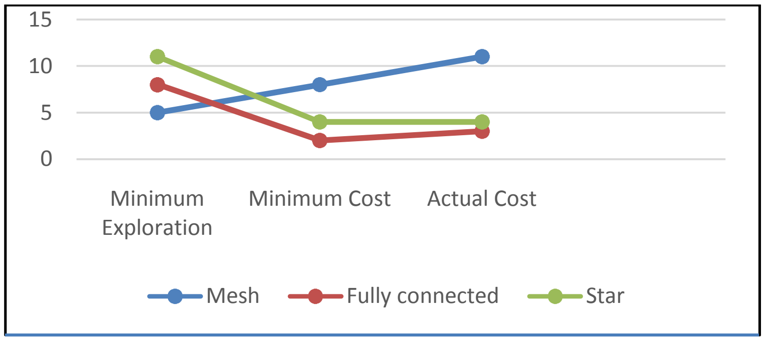


Figure14. How the exploration time is changing when changing algorithm

Table 2. Number of steps needed in different topology applied in different algorithm

Network Topology	Minimum Exploration	Minimum Cost	Actual Cost
Star Network	5	8	11
Fully-connected Network	8	2	3
Mesh Network	11	4	4

The above table shows how the exploration time changes with different routing algorithm. With star network, the best routing algorithm to use is heuristic algorithm because it has the fewest steps to discover the routing path. In mesh network, time needed to find the best routing path with cost-minimizing algorithm and dijkstra's algorithm are same.

8. Conclusion

In this paper, we have reported the results of our programming model to find the routing path in WSNs and investigated the performance of our model when applied in different routing algorithms and multiple network topologies and further scope for improvement. The goal of the research is to provide a new programming model that can be applied in different network topologies to deliver the most accurate result. Research must carry on in all the capacities of WSN programming model. Still there are some qualities and features are missing to let WSN programming model reach its best level of performance. However, we believe that WSN programming model is a huge step towards deploy more applications in WSNs domain.

REFERENCES

[1] Rong Yu, Zhi Sun, and Shunliang Mei, "Scalable Topology and Energy Management in Wireless Sensor Networks", presented at the Wireless Communication and Networking Conference (WCNC) 2007.IEEE

[2] Tubaishat, M. and Madria, S.K., " Sensor networks: an overview", Potentials, IEEE, On page(s): 20 – 23 Vol 22 (2), April-May 2003

[3] Marriwala, N. and Rathee, P. ," An approach to increase the wireless sensor network lifetime", Information and Communication Technologies (WICT), 2012 , IEEE, On page(s): 495 - 499 Oct. 30 2012-Nov. 2 2012

[4] R. Newton and M. Welsh. “Region streams: functional macroprogramming for sensor networks”. In Proceedings of DMSN’04, pages 78–87, New York, NY, USA, 2004. ACM Press.

[5] Supasate, C. Nuttanart,P. and Chalermek,I. “Logic Macroprogramming for Wireless Sensor Networks” International Journal of Distributed Sensor Networks, 2012.

[6] Bischoff, U. and Kortuem, G., “A State-based Programming Model and System for Wireless Sensor Networks”, In: PerCom Workshops 2007 - Fifth Annual IEEE International Conference on Pervasive Computing and Communications, On page(s) 19-23 March, 2007

[7] Alajlan, Abrar.,et al., “Topology Management In Wireless Sensor Networks: Multi-State Algorithms”, International Journal of Wireless & Mobile Networks (IJWMN), On page(s) 17-26 Vol 4 (6), Dec 2012

[8] Vardhe, K., Chi Zhou; Reynolds, D., ” Energy Efficiency Analysis of Multistage Cooperation in Sensor Networks” Global Telecommunications Conference (GLOBECOM 2010), IEEE, On page(s) 1-5 Dec, 2010

[9] Sousa,M.P., et al., “Scalability in an Adaptive Cooperative System for Wireless Sensor Networks” International Conference on Ultra Modern Telecommunications & Workshops (ICUMT 2009) IEEE, On page(s) 1 – 6, Oct 12

[10] Warneke,B., et al., “Autonomous Sensing and Communication in A Cubic Millimeter “IEEE, On page(s) 44 – 51, Vol 34 (1) Jan 2001.

[11] Takizawa, Y., “Node Localization for Sensor Networks using Self-Organizing Maps”, Wireless Sensors and Sensor Networks (WiSNet), IEEE, On page(s) 61 - 64 , Jan 2011.

[12] Pandey, S., et al., “Localization of Sensor Networks Considering Energy Accuracy Tradeoffs” Collaborative Computing: Networking, Applications and Worksharing, International Conference, IEEE, On page(s) 1 - 10, 2005.

[13] Qiang Liu., et al., “Time Synchronization Performance Analysis and Simulation of a kind of wireless TDMA Network” International Frequency Control Symposium and Exposition, 2006 IEEE, On page(s) 299 – 303, June 2006.

[14] Yaguang K., Xifang Z., Huakui C., “ Intelligent Time Synchronization in Sensor Network”, Wireless, Mobile and Multimedia Networks, 2006 IET International Conference,IEEE, On page(s) 1 - 4 , Nov. 2006.

[15] Lasassmeh, S.M., Conrad, J.M., “ Time Synchronization in Wireless Sensor Networks: A Survey”, IEEE SoutheastCon 2010 (SoutheastCon), On page(s) 242 - 245 , March 2010.

[16] Liming He, Geng-Sheng Kuo .,” A Novel Time Synchronization Scheme in Wireless Sensor Networks”, Vehicular Technology Conference, 2006. VTC 2006 - spring. IEEE 63rd, On page(s) 568 - 572, Vol: 2, May 2006.

[17] Ryo Sugihara , Rajesh K. Gupta., “Programming Models for Sensor Networks: A Survey”, ACM Transactions on Sensor Networks, vol. 4, no. 2, article no. 8, 2008.

[18] Gupta, V., et al., “Nano-CF: A coordination framework for macro-programming in Wireless Sensor Networks”, Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2011 8th Annual IEEE Communications Society, On page(s) 467 – 475, June 2011.

[19] Supasate C., Nuttanart P., Chalermek I., “ Logic Macroprogramming For Wireless Sensor Networks”, International Journal of Distributed Sensor Networks, On page(s) 1-12, Vol. 2012, April 2012.

[20] R. Gummadi, O. Gnawali, and R. Govindan. Macroprogramming wireless sensor networks using Kairos. In Intl. Conf. Distributed Computing in Sensor Systems (DCOSS), June 2005.

[21] Supasate C., Chalermek I., "An Analysis of Deductive-Query Processing Approaches for Logic Macroprograms in Wireless Sensor Networks", Engineering Journal, Vol. 16, No. 4, July 2012.

[22] L. Mottola and G.P. Picco, "Programming wireless sensor networks: Fundamental concepts and state of the art", ;presented at ACM Comput. Surv., 2011, On page(s) 19-19.

[23] R. Gummadi, N. Kothari, T. Millstein, and R. Govindan, “Declarative failure recovery for sensor networks,” in Proc. Int. Conf. Aspect-oriented software development. ACM, Mar. 2007, pp. 173–184.

[24] Newton, R. and Welsh, M., “Region streams: Functional macro-programming for sensor networks”. In Proceedings of the 1st International Workshop on Data Management for Sensor Networks. 2004.

[25] Moggi, E., “Computational lambda-calculus and Monads”, Logic in Computer Science, 1989. LICS '89, Proceedings., IEEE, On page(s): 14 – 23, Jun 1989.

[26] Newton, R., Morrisett, G.; Welsh, M., “The Regiment Macroprogramming System”, Information Processing in Sensor Networks, 2007, IEEE, On page(s): 489 - 498 , April 2007.

[27] S. Madden, et al., "TinyDB: an acquisitional query processing system for sensor networks", ACM Transactions on Database Systems, On page(s): 122-173, Vol: 30 (1), March 2005.
[28] James Horey, Eric Nelson, Arthur B. Maccabe., "Tables: A Spreadsheet-Inspired Programming Model for Sensor Networks", DCOSS'10 Proceedings of the 6th IEEE international conference on Distributed Computing in Sensor Systems, On page(s): 1-14, 2010.
[29] W. F. Fung, D. Sun, and J. Gehrke, "Cougar: the network is the database," In SIGMOD Conference, pp. 621, 2002.

Authors

Mrs. Abrar Alajlan: is a Ph.D. student department of Computer Science and Engineering at the University of Bridgeport, Bridgeport, CT. She earned a master's degree in MBA with a concentration in Information Systems (IS) from Troy University, Troy, AL in 2011. She received a BS in Computer Science from Umm Al-Qura University, Makkah, Saudi Arabia. Abrar's interests are in Wireless Sensor Network (WSN), Network Security, Mobile Communication.

Dr. Khaled Elleithy: is the Associate Dean for Graduate Studies in the School of Engineering at the University of Bridgeport. His research interests are in the areas of, network security, mobile wireless communications formal approaches for design and verification and Mobile collaborative learning. He has published more than two hundreds research papers in international journals and conferences in his areas of expertise.
Dr. Elleithy is the co-chair of International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering (CISSE).CISSE is the first Engineering/ Computing and Systems Research E-Conference in the world to be completely conducted online in real-time via the internet and was successfully running for four years. Dr. Elleithy is the editor or co-editor of 10 books published by Springer for advances on Innovations and Advanced Techniques in Systems, Computing Sciences and Software.
Dr. Elleithy received the B.Sc. degree in computer science and automatic control from Alexandria University in 1983, the MS Degree in computer networks from the same university in 1986, and the MS and Ph.D. degrees in computer science from The Center for Advanced Computer Studies in the University of Louisiana at Lafayette in 1988 and 1990, respectively. He received the award of "Distinguished Professor of the Year", University of Bridgeport, during the academic year 2006-2007.

Mr. Varun Pande is a Graduate Research Assistant currently attending the University of Bridgeport as a PhD candidate in Computer Science and Engineering. He graduated from the University of Bridgeport with a Master in computer Science in May of 2012. He had worked as a CSR representative at TATA Power, during his Bachelor in computer science and Information Technology. Currently and for the past two years he has been a Graduate Assistant and taught Labs on Wireless Sensor Communication using MICA z Motes. His research interests are Computer Vision, Image Processing, Parallel processing and Wireless Sensor Networks. He hopes to share my experiences, research and knowledge with other graduates and professionals to work in a collaborative research for a Better tomorrow!