IJCAT International Journal of Computing and Technology, Volume 1, Issue 2, March 2014
ISSN : 2348 - 6090
www.IJCAT.org

Implementation of Least-Latency Routing over Time-Dependent Wireless Sensor Networks

[1] **Manju Priya G**

[1] Computer Science, Visvesvaraya Technological University
Kolar Gold Fields, Karnataka ,India

Abstract - We consider the problem of least-latency end-to-end routing over adaptively duty-cycled wireless sensor networks. Such networks exhibit a time-dependent feature, where the link cost and transmission latency from one node to other nodes vary constantly in different discrete time moments. We model the problem as the time-dependent Bellman-Ford problem. We show that such networks satisfy the first-in-first-out (FIFO) property, which makes the time-dependent Bellman-Ford problem solvable in polynomial-time. Using the beta-synchronizer, we propose a fast distributed algorithm to construct all-to-one shortest paths with polynomial message complexity and time complexity. The algorithm determines the shortest paths for all discrete times in a single execution, in contrast with multiple executions needed by previous solutions. We further propose an efficient distributed algorithm for time-dependent shortest path (TDSP) maintenance. We discuss a suboptimal implementation of our proposed algorithms that reduces their memory requirement.

Keywords **- Time dependent, shortest path, wireless sensor networks, routing, routing maintenance, least latency.**

1. Introduction

MULTIHOP data routing over wireless sensor networks (WSNs) has attracted extensive attention in the recent years. Since there is no infrastructure in sensor networks, the routing problem is different from the one in traditional wired networks or the Internet. Some routing protocols [2], [3] over WSNs presented in the literature are extended from the related approaches over wired/wireless ad hoc networks. They usually find a path with the minimum hop count to the destination, which is based on the assumption that the link cost (or one-hop transmission latency) is relatively static for all wired/wireless links. However, for duty-cycled WSNs [4], [5], [6], that assumption may not always be valid. Duty-cycled WSNs include sleep-wake up mechanisms, which can violate the assumption of static link costs. Currently, many MAC protocols support WSNs operating with low duty cycle, e.g., B-MAC [5], X-MAC [6]. In such protocols, sensor nodes operate in low power listening (or LPL) mode. In the LPL mode, a node periodically switches between the active and sleep states. The time duration of an active state and an immediately following sleep state is called the LPL checking interval, which can be identical, or can be adaptively varied for different nodes, referred to as ALPL [7]. The duty-cycled mechanism has been shown to achieve excellent idle energy savings, scalability, and easiness in implementation. However, they suffer from time-varying neighbour discovery latencies (the time between data arrival and discovery of the adjacent receiver), which is also pointed out by Gu and He [8]. As shown in Fig. 1, the neighbour discovery latency is varying with different departure times. Even with synchronized duty cycling, the neighbour discovery latency is varying at different time moments due to adaptive duty cycle setting as shown in Fig. 1. To formally define the problem, we first define the link Cost as the time delay between data dispatching time, which is the earliest time when a sender wakes up for data transmission, and the data arrival time at the receiver. The link cost is time-varying in adaptively duty-cycled WSNs due to varying neighbour discovery latencies, even though the physical propagation condition does not change with time.

The dispatching time is the time moment when the data is ready for transmission at the sender side. Thus, this raises a nontrivial problem: with time-varying link costs, how to find optimal paths with least nodes-to-sink latency for all nodes at all discrete dispatching time moments? A similar problem has been modelled in previous works as the time-dependent shortest path problem (or TDSP) [9], [10] in the field of traffic networks [11], time-dependent graphs [12], and GPS navigation [13]. The general time dependent shortest path problem is at least NP-Hard, since it may be used to solve a variety of NP-Hard optimization problems such as the knapsack problem. However, depending on how one defines the problem, it may not be in NP, since its output is not polynomially bounded. Moreover, there are even continuous-time instances of the TDSP problem in which shortest paths consist of an infinite sequence of arcs, as shown by Orda and Rom [14]. In this paper, we study a special case where the networks are known as first-

in-first-out (FIFO) networks, in which commodities travel along links in a First-In-First-Out manner. Under the FIFO condition, the time-dependent shortest path problem is solvable in polynomial time. The TDSP problem has also been studied with a distributed approach. The only previous distributed solution [10] computes the shortest paths for a specific departure time in each execution. If the whole time period has M discrete intervals (M is 1 for infinite time intervals),we have to execute the algorithm in [10] M times, which is inefficient in terms of message complexity and time complexity, given the limited power and radio resource in WSNs. Therefore, the first motivation of our work is to design a fast distributed algorithm for the problem, which can efficiently enumerate all optimal paths with least end-to-sink latency for infinite time intervals.

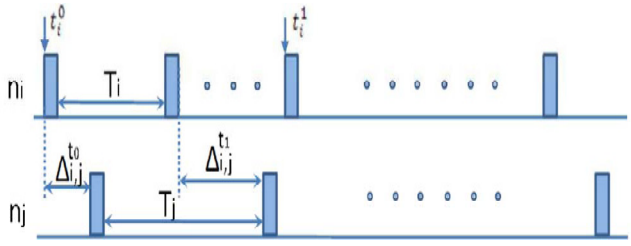

Fig.1. Varying neighbour discovery latency in heterogeneous LPL mode

The second motivation of our work is to propose algorithms which can dynamically and distributive maintain time dependent least-latency paths. In WSNs, a node may update its duty-cycle configuration (e.g., based on residual energy), or join or leave the network, thereby changing the network topology. In such situations, the duty-cycle updating node or the joining/leaving node may change the cost of all the links with its neighbours, which means that a single node update causes multiple link updates. Previous efforts on this problem [15], [16] are efficient in handling single link updates. However, applying such solutions for multiple link updates would imply that multiple distributed updates execute concurrently for a single node update, which is not efficient in terms of message cost and memory cost for resource-limited WSNs. The third motivation is to address practical implementation issues. Our work requires schedule-awareness in neighbourhood. One scenario for such requirement is that all nodes have none information from each other after initial deployment in field. In this scenario, we would like to get schedule-awareness without global time synchronization. Local synchronization and light information exchange is desirable to get duty-cycling information and time difference from neighbours for a sensor node. One candidate to achieve this is by neighbour discovery protocol, just similar as the link layer neighbour discovery protocol in the Internet. Nodes without schedules of neighbours would stay awake and broadcast neighbour discovery message periodically (i.e., once every multiple predefined time slots) and neighbours can response back their schedule information, which is a kind of local level synchronization. Once reaching a stable stage, any node would get its neighbour's schedule and the time difference from its neighbours, and switch back to duty-cycling state. The update of nodes schedule can be undergoing in background either proactively or reactively. Because our work is not limited to any specified MAC protocol, we discuss different methods to achieve schedule awareness over several underlying mechanisms, such as B-MAC, S-MAC, and quorum-based wakeup scheduling. Regarding to another practical implementation issues, we also need to understand how to simplify the vector presentation so that only smaller vector sizes are required, given the limited memory resource of sensor nodes. We present a suboptimal implementation, which achieves tradeoffs between latency and memory usage.

Finally, we discuss the complexities of our algorithms in some special scenarios, like static link costs and multiple sink nodes. In this paper, we first propose a distributed algorithm to compute the time-dependent paths with least latency for all nodes in a duty-cycled WSN. The algorithm has low message and space complexities. The algorithm is based on the observation that the time-varying link cost function is periodic, and hence by derivation, the time-varying distance function for each node is also periodic. We show that the link cost function satisfies the FIFO property [9]. Therefore, the time-dependent shortest path problem is not an NP-hard problem, and thus is solvable in polynomial time. We also propose distributed algorithms for maintaining the shortest paths. The proposed algorithms recomputed the routing paths based on previous path information.

The contributions of the paper are as follows:

1. We model adaptively duty-cycled WSNs as time-dependent networks. We show that such networks satisfy the FIFO condition and the triangular path condition.
2. We present distributed algorithms for finding the time-dependent shortest paths to the sink node for all nodes. When compared to the previous solution [10], our algorithms find the shortest paths in a single execution for infinite time intervals.
3. We present distributed shortest path maintenance algorithms with low message complexity and space complexity.
4. We propose suboptimal implementation with vector compression. To the best of our knowledge, we are not aware of any other efforts that consider duty-cycled WSNs as time-dependent networks and solve the problem of

finding or updating the shortest paths with efficient message and space costs.

2. Related Work

We summarize the literature on LPL scheduling and the time-dependent shortest path problem as follows. LPL/ALPL in WSNs. LPL means that a node only wakes up and listens the channel state for a short time period. Examples include B-MAC [5] which is a CSMA-based technique utilizing low power listening and an extended preamble to achieve low power communication. In B-MAC, nodes have an independent sleep-wake up schedule. If a node wishes to transmit, it precedes the data packet with a preamble that is slightly longer than the sleep period of the receiver. During the wake period, a node samples the medium, and if a preamble is detected, it remains awake to receive the data. With the extended preamble, a sender is assured that at some point during the preamble, the receiver will wake up, detect the preamble, and remain awake in order to receive the data. The designers of B-MAC show that B-MAC surpasses existing protocols in terms of throughput, latency, and for most cases, energy consumption. While B-MAC performs quite well, it suffers from the overhearing problem, and the long preamble dominates the energy usage. To overcome some of B-MAC's disadvantages, XMAC [6] and DPS-MAC [17] were proposed, In XMAC or DPSMAC, short preamble was proposed to replace the long preamble in B-MAC. Also, receiver information is embedded in the short preamble to avoid the overhearing problem. The main disadvantage of B-MAC, X-MAC, and DPS-MAC is that it is difficult to reconfigure the protocols after deployment, thus lacking in flexibility. X-MAC [6] and DPS-MAC [17] are compatible with LPL mechanisms.

However, they do not explicitly support adaptive duty cycling, where nodes choose their duty cycle depending on their residual energy. Raja et al. [7] and Vigorito et al. [4] present adaptive low power listening (ALPL) mode based on nodes' residual energy. These works provide the application spaces for our work. In ALPL, since nodes have heterogeneous duty cycle setting, it is more difficult for neighbour discovery since a node cannot differentiate whether a neighbour is sleeping or failing when it does not receive feedback from the neighbour.

ALPL also incurs time-dependent link-cost and end-to-end latency as illustrated in Section 4. Recently, B-MAC [5] was also extended to support the ALPL mode in TinyOS.

Delay-efficient routing over adaptively duty-cycled WSNs. Over adaptively duty-cycled WSNs, routing becomes more difficult due to two reasons: intermittent connection between two neighbour nodes and changes in the transmission latency at different times. Some works have studied the delay-efficient routing problem over adaptively duty-cycled WSNs in recent years. Su et al. [18] proposed two methods to solve routing over intermittently connected WSNs due to duty cycling. One is by an on-demand approach that uses probe messages to determine the least-latency route. The other one is a proactive method, where all least-latency routes at different departure times are computed at the beginning. The first method does not work well for frequent data deliveries. The second method is a centralized approach, and is not flexible for distributed construction. Our algorithms also follow the proactive approach, but are distributed.

Time-dependent shortest path problem. This problem was first proposed by Cooke and Halsey [9]. It has been well studied in the field of traffic networks [11], time-dependent graphs [12], and GPS navigation [13]. Previous solutions for this problem mostly work offline using a centralized approach [12]. Although these solutions can provide inspirations, they cannot be applied to WSNs where the global network topology is not known by a centralized node, given the large-scale size of a WSN. For the distributed time-dependent shortest path problem, the only previous work [10] computes the shortest paths for a specific departure time in each execution, which is not time efficient. If the whole time period has M discrete intervals (M is 1 for infinite time intervals), we have to execute the algorithm in [10] M times, which is inefficient in terms of message complexity and time complexity. For multiple executions, the algorithm in [10] suffers from high message cost, which is undesirable for resource-limited WSNs. The work in [10] discusses two policies for the time-dependent shortest path problem: waiting and non-waiting.

Waiting does not mean waiting in the buffer, but means waiting for some time after the data has been delivered (i.e., the receiver is awake). Non-waiting means that a sender will immediately send the data once the receiver is awake. We do not consider the waiting policy in our work, since the end-to-end latency does not benefit from waiting.

Dynamic shortest path maintenance. Many works [15], [16], [23] exist for handling link decreases and increases, an algorithm is given for computing all-pairs shortest paths, which requires $O(n^2)$ messages when the network size is n. In [25], an efficient incremental solution has been proposed for the distributed all-pairs shortest paths problem, requiring O(nlog(nW)) amortized number of messages over a sequence of edge insertions and edge weight decreases. Here, W is the largest positive integer edge weight. In [26], Awerbuch et al. propose a general technique that allows to update the all-pairs shortest paths in a distributed network in O(n) amortized number of messages and O(n) time, by using $O(n^2)$ space per node.

β-Synchronizer [27]. As described in [27], the synchronizer is a methodology for designing efficient distributed algorithms in asynchronous networks. Researchers have used synchronizers to reduce message complexity of some asynchronous algorithms, such as Bellman-Ford. A synchronizer works as follows: a synchronizer generates sequences of "clock-pulses" at each node of a network. At each node, a new pulse is generated only after it receives all the messages which were sent to that node by its neighbours at the previous pulse. Thus, a synchronizer runs in a phase- by- phase manner.

A β-synchronizer is a special type of synchronizer, which has an initialization phase, in which a leader s is chosen in the network and a spanning tree rooted at s is constructed (e.g., by a Breadth-First-Search). After the execution of one phase, the leader s will eventually learn that all the nodes in the network are "safe." At that time, it broadcasts a message along the spanning tree, notifying all the nodes that they may generate a new pulse. The communication pattern for receiving all acknowledgments is just like convergecast. Therefore, with a β-synchronizer, whenever a node learns that it is safe and all its descendants in the tree are safe, it sends an acknowledgment to its parent.

3. Assumptions and Problem Definition

The wakeup schedule depends on the underlying MAC protocol. We first assume that a node can be operated in the LPL mode: a node wakes up at the beginning of a time slot to check the channel state. If there is no activity, the node goes back to sleep; otherwise, it stays awake. Then, we relax the assumption and discuss how our work can be applied to other wakeup schedules, such as quorum schedules [28]. We consider the non-waiting policy (i.e., the sender immediately delivers data once the receiver is awake) at each node, since the node-to-sink delay will not benefit from waiting. Thus, once the data arrives at an intermediate node, the node will attempt to dispatch the data immediately. Dispatching time represents the earliest time when a sender is awake for data transmission. Thus, dispatching times are not the same as the data departure times, as the data may still be buffered in the sender's memory. The general problem of determining the shortest paths with the least latency in time-dependent WSNs can be defined as follows: Find the least-time paths from all nodes to the sink node ns corresponding to the minimum achievable delay.

$$d_i(t_i^k)\text{-}\min_{nj\in Ni}\{\ \tau_{i,j}(t_i^k) + d_j(t_i^k + \tau_{i,j}(t_i^k))\ \} \qquad (1)$$

Equation (1) is an extension of Bellman's equations [29] for the time-dependent network and is referred to as TD-Bellman's equation hereafter.

4. Modelling Adaptively Duty – Cycled WSNS

We will now show that the link cost function is periodic and establish that the time-varying distance function is also periodic. Having done so, we will show how TD-Bellman's Equation can be implemented by vector representations of link costs and distances.

A. Link Cost Function

Without loss of generality, suppose there are two adjacent nodes ni and nj, where ni is the sender and nj is the receiver.

B. Distance to Sink

We refer to the node-to-sink delay as distance, for compatible representation with that in the static Bellman-Ford algorithm [29].

C. Implementation via Vectors

We implement the discrete, periodic, and infinite link cost functions and the distance functions as vectors, and implements the TD-Bellman's equation by vector operations. Our goal is to use vectors with limited sizes in order to implement our algorithm with limited memory and same message size over the air, although the time axis is infinite.

5. Algorithm for Initial Route Construction

We now present an algorithm for initial time-dependent shortest path route construction in duty-cycled WSNs, where the distances from all nodes to the sink node are initially infinite. The proposed algorithm, referred to as the FTSP algorithm, for Fast Time-Dependent Shortest Path algorithm, is inspired by the work in [10].Although the time axis is infinite, the time-varying link costs and distance can be implemented by vectors. Therefore, our algorithm which implements (1) is basically similar to the distributed Bellman-Ford algorithm. The difference is that our algorithm is exchanging vectors (i.e., for time-varying link costs and distances), rather than single values of static link cost and distance.

Distributed algorithm for route construction

There are essentially two steps in our algorithm for constructing routing paths. In the first step, we build a spanning tree. In the second step, we calculate the shortest paths and send back the acknowledgment to the sink for nodes in the network with a layer-by-layer approach .In our algorithm FTSP, we combine the two steps and

implement them through iterations. In the first iteration, FTSP computes the shortest paths for nodes in the nearest layer to the sink. In the following iterations, FTSP goes beyond one layer each time, until it reaches the last layer.

6. Algorithm for Dynamic Route Maintenance

When compared with static networks, link changes and node changes are more frequent in duty-cycled WSNs. If a node changes its duty-cycle configuration, or dynamically joins or leaves the network, the links connecting with all its neighbours will be changed at multiple time intervals. In such a situation, a single node update usually causes multiple link updates.

We propose a solution in which a node only stores the route to the sink, which is more practical in WSNs due to their memory constraints. In our proposed algorithm, when one node is updated (denoted as the source node), the algorithm does not update the shortest path for the whole network from scratch, but only updates necessary nodes. Thus, the main idea is first to identify which nodes need to be updated. After that, the algorithm updates the shortest path for these identified nodes. The updating process is similar to the route construction. But the starting point is the source node, rather than from the sink node and the updating scope is just a subset of the whole network.

7. Schedule Awareness

FTSP require awareness of wakeup schedules of the neighbourhood. Achieving this requirement depends on the specific underlying MAC protocol. We chose two MAC protocols: ALPL and quorum-based duty-cycling.

In the **ALPL mode**, a node just wakes up for a short time during a checking interval to check the channel activities. The duration of the checking interval varies for different nodes. We changed the duration of the checking interval in our simulation experiments with four sets, C1, C2, C3, and C4, as listed in Table1. With each set, we randomly chose one element as the value of the LPL checking interval for each node. With different time slot sets, the size of a message is changing. Thus, we used a flexible packet size in our simulation. Each element in a vector occupied 1 byte in all experiments.

Table I
Time Slot Sets

C_1 (ms)	$\{100, 100, 100, 100\}$
C_2 (ms)	$\{100, 200, 300, 600\}$
C_3 (ms)	$\{100, 200, 400, 800\}$
C_4 (ms)	$\{100, 200, 500, 1000\}$

Quorum-based duty-cycling: An active neighbour discovery mechanism requires a node to stay awake actively. Now we introduce quorum-based duty-cycling which does not have that requirement. Here, the wakeup schedule follows a quorum system design. In quorum-based duty cycling, two neighbour nodes can hear each other at least once within bounded time slots via the nonempty intersection property.
For quorum-based duty-cycling, we choose the {7; 3; 1} and {21; 5; 1} difference sets for the heterogeneous wakeup schedule settings. The duration of one time slot was set to 100 ms in quorum-based duty-cycling. Since FTSP, FTSP-M are independent of wakeup scheduling, we argue that the comparison is fair even when we choose quorum-based duty-cycling.

8. Suboptimal Implementation with Vector Compression

The key implementation aspect of our proposed algorithms is the vector representation of link cost functions and distance functions. However, if the vector size is too large (i.e., the LCM is too large), the proposed algorithms, FTSP and FTSP-M, may not be feasible given the limited memory resource of embedded sensor nodes, and inefficient due to distributed message exchanging.
In a real implementation, to avoid arbitrarily long vectors, there are two possible solutions: use a **predefined duty cycle** set, so that they $LCM\ (T0, T1, \ldots..Ti) \frac{}{} Ti$ can be bounded by carefully selecting a duty cycle, adopt **vector compression** to achieve a tradeoffs, i.e., adopt a low-accurate distance vector, which takes less memory space, to represent the end-to-end latency. Hence, the output path is suboptimal in terms of latency. The first solution can be applied to small-scale networks, where the node number is not large and predefining a duty cycle set is not difficult. For a large-scale network, we might need the second solution in which a bounded, global $LCM\ (T0, T1, \ldots..Ti) \frac{}{} Ti$ is not necessary. The basic idea of vector compression in the second solution is to smooth all values in a vector and represent the vector with less information.

9. Conclusions

In this paper, we addressed the distributed shortest path routing problem in duty-cycled WSNs. Our contributions are four-fold. First, we modelled duty-cycled WSNs as time dependent networks, which satisfy the FIFO condition. Second, we presented the FTSP algorithm for finding shortest paths in such networks. FTSP has polynomial message complexity and is more time-efficient

than previous solutions. Third, we presented FTSP-M for distributed route maintenance with node insertion, updating, and deletion. FTSP-M is memory efficient and has polynomial message complexity. Finally, we proposed a suboptimal implementation on vector representations to reduce memory requirements. The vector size of the suboptimal solution does not depend on the largest LCM value. We envision several directions for future work. One is to investigate the time-dependent minimum spanning tree problem, which is NP-Hard in duty-cycled WSNs. Another direction is to study time-dependent multicast routing in duty-cycled WSNs, which is a required service for many applications and is the reverse direction of all-to-one least latency routing.

References

[1] S. Lai and B. Ravindran, "On Distributed Time-Dependent Shortest Paths over Duty-Cycled Wireless Sensor Networks," Proc. IEEE INFOCOM, 2010.

[2] C. Schurgers and M.B. Srivastava, "Energy Efficient Routing in Wireless Sensor Networks," Proc. Military Comm. Conf. (MILCOM 01), vol. 1, pp. 357-361, 2001.

[3] A. Woo, T. Tong, and D. Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks," Proc. First Int'l Conf. Embedded Networked Sensor Systems (Sensys), pp. 14-27, 2003.

[4] C.M. Vigorito, D. Ganesan, and A.G. Barto, "Adaptive Control of Duty Cycling in Energy-Harvesting Wireless Sensor Networks," Proc. IEEE Comm. Soc. Conf. Sensor, Mesh and Ad Hoc Comm. And Networks (SECON), pp. 21-30, June 2007.

[5] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," Proc. Int'l Conf. Embedded Networked Sensor Systems (Sensys '04), pp. 95-107, 2004.

[6] M. Buettner, G.V. Yee, E. Anderson, and R. Han, "X-mac: A Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks," Proc. ACM Fourth Int'l Conf. Embedded Networked Sensor Systems (Sensys), pp. 307-320, 2006.

[7] J. Raja, B. Pierre, and V. Cristina, "Adaptive Low Power Listening for Wireless Sensor Networks," IEEE Trans. Mobile Computing,vol. 6, no. 8, pp. 988-1004, Aug. 2007.

[8] Y. Gu and T. He, "Data Forwarding in Extremely Low Duty-Cycle Sensor Networks with Unreliable Communication Links," Proc. Sixth ACM Conf. Embedded Network Sensor Systems (Sensys), pp. 32-38, 2007.

[9] K.L. Cooke and E. Halsey, "The Shortest Route Through a Network with Time-Dependent Internodal Transit Times," J. Math. Analysis Application, vol. 14, pp. 493-498, 1966.

[10] A. Orda and R. Rom, "Distributed Shortest-Path Protocols for Time-Dependent Networks," Distributed Computing, vol. 10, no. 1, pp. 49-62, 1996.

[11] I. Chabini, "Discrete Dynamic Shortest Path Problems in Transportation Applications: Complexity and Algorithms with Optimal
Run Time," Transportation Research Records, vol. 1645, pp. 170-175,
1998.

[12] B. Ding, J.X. Yu, and L. Qin, "Finding Time-Dependent Shortest Paths over Large Graphs," Proc. 11th Int'l Conf. Extending Database Technology (EDBT '08), pp. 205-216, 2008.

[13] H. Chon, D. Agrawa, and A. Abbadi, "Fates: Finding a Time Dependent Shortest Path," Mobile Data Management, vol. 2574, pp. 165-180, 2003.

[14] A. Orda and R. Rom, "Minimum Weight Paths in Time-Dependent Networks," Networks, vol. 21, pp. 295-319, 1991.

[15] C. Serafino, D. Gabriele, F. Daniele, and N. Umberto, "A Fully Dynamic Algorithm for Distributed Shortest Paths," Theoretical Computer Science, vol. 297, nos. 1-3, pp. 83-102, 2003.

[16] G. D'Angelo, S. Cicerone, G.D. Stefano, and D. Frigioni, "Partially
Dynamic Concurrent Update of Distributed Shortest Paths," Proc. Int'l Conf. Computing: Theory and Applications, pp. 32-38, 2007.

[17] H. Wang, X. Zhang, F. Abdesselam, and A. Khokhar, "DPS-MAC:An Asynchronous MAC Protocol for Wireless Sensor Networks," Proc. 14th Int'l Conf. High Performance Computing, vol. 7, pp. 393-404, June 2007.

[18] L. Su, C. Liu, H. Song, and G. Cao, "Routing in Intermittently Connected Sensor Networks," Proc. IEEE Int'l Conf. Network Protocols, pp. 278-287, 2008.

[19] Y. Gu and T. He, "Bounding Communication Delay in Energy Harvesting Sensor Networks," Proc. IEEE 30th Int'l Conf. Distributed Computing Systems, pp. 837-847, 2010.

[20] Y. Gu, T. He, M. Lin, and J. Xu, "Spatiotemporal Delay Control for Low-Duty-Cycle Sensor Networks," Proc. IEEE 30th Real-Time Systems Symp. (RTSS), pp. 127-137, 2009.

[21] X. Yang and N.H. Vaidya, "A Wakeup Scheme for Sensor Networks: Achieving Balance between Energy Saving and End-to- End Delay," Proc. IEEE 10th Real-Time and Embedded Technology and Applications Symp., 2004.

[22] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel, "Delay Efficient Sleep Scheduling in Wireless Sensor Networks," Proc. IEEE INFOCOM, pp. 2470-2481, 2005.

[23] K.V.S. Ramarao and S. Venkatesan, "On Finding and Updating Shortest Paths Distributively," J. Algorithms, vol. 13, no. 2, pp. 235- 257, 1992.

[24] S. Haldar, "An "All Pairs Shortest Paths" Distributed Algorithm Using 2n2 Messages," J. Algorithms, vol. 24, no. 1, pp. 20-36, 1997.

[25] G.F. Italiano, "Distributed Algorithms for Updating Shortest Paths (Extended Abstract)," Proc. Fifth Int'l Workshop Distributed Algorithms (WDAG), pp. 200-211, 1992.

[26] B. Awerbuch, I. Cidon, and S. Kutten, "Communication-Optimal Maintenance of Replicated Information," Proc.

31st Ann. Symp. Foundations of Computer Science (SFCS), pp. 492-502, 1990. [27] B. Awerbuch, "Complexity of Network Synchronization," J. ACM, vol. 32, no. 4, pp. 804-823, 1985.

[28] S. Lai, B. Zhang, B. Ravindran, and H. Cho, "CQS-Pair: Cyclic Quorum System Pair for Wakeup Scheduling in Wireless Sensor Networks," Proc. Int'l Conf. Principles of Distributed Systems (OPODIS). pp. 295-310, 2008.

[29] R. Bellman, "On a Routing Problem," Quarterly of Applied Math., vol. 16, no. 1, pp. 87-90, 1958.

[30] P. Sommer and R. Wattenhofer, "Gradient Clock Synchronization in Wireless Sensor Networks," Proc. ACM Int'l Conf. Information Processing in Sensor Networks (IPSN), pp. 37-48, 2009.

[31] K.M. Chandy and J. Misra, "Distributed Computation on Graphs: Shortest Path Algorithms," Comm. ACM, vol. 25, no. 11, pp. 833- 837, 1982.

[32] A. Segall, "Distributed Network Protocols," IEEE Trans. Information Theory, vol. 29, no. 1, pp. 23-35, Jan. 1983.

[33] J.J. Garcia-Lunes-Aceves, "Loop-Free Routing Using Diffusing Computations," IEEE/ACM Trans. Networking, no. 1, pp. 130-141, Feb. 1993.

[34] Y. Sun, O. Gurewitz, and D.B. Johnson, "RI-MAC: A Receiver- Initiated Asynchronous Duty Cycle MAC Protocol for Dynamic Traffic Loads in Wireless Sensor Networks," Proc. ACM Conf. Embedded Network Sensor Systems (Sensys), pp. 1-14, 2008.

[35] W.S. Luk and T.T. Huang, "Two New Quorum Based Algorithms for Distributed Mutual Exclusion," Proc. Int'l Conf. Distributed Computing Systems (ICDCS), pp. 100-106, 1997.

[36] W. Ye, J. Heidemann, and D. Estrin, "Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks," IEEE/ACM Trans. Networking, vol. 12, no. 3, pp. 493-506, June 2004.

[37] T.V. Dam and K. Langendoen, "An Adaptive Energy-Efficient Mac Protocol for Wireless Sensor Networks," The First ACM Conf. Embedded Networked Sensor Systems (Sensys), 2003.

[38] S. Mallat, A Wavelet Tour of Signal Processing: The Sparse Way, third ed. Academic Press, 2008.

[39] OMNET++, http://www.omnetpp.org/, 2012.

[40] M. Zuniga and B. Krishnamachari, "Analyzing the Transitional Region in Low Power Wireless Links," Proc. IEEE Comm. Soc. Conf. Sensor, Mesh and Ad Hoc Comm. and Networks (SECON), pp. 517- 526, 2004.

[41] C.E. Perkins and P. Bhagwat, "Highly Dynamic Destination- Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," ACM SIGCOMM Computer Comm. Rev., vol. 24, no. 4, pp. 234-244, 1994.

Ms Manju Priya G, Student of M.tech Final Year studying in Atria Institute of Technology, Bangalore .She received the B.E degree in Computer Science and Engineering, from HKBKCE, Bangalore in 2011. She has received progressive success in various Debate Competition and Elocution. Her main research interests include network securities, Cryptography, Database Management, and Image Processing.