Optimal Duty Cycling with Sleep-wake Schedule between Paired Nodes and Flexible Routing across Pairs

Ginish
Research Scholar (ECE)
SBSSTC, Ferozepur

Munish Kumar
Assistant professor(ME)
SBSSTC, Ferozepur

Amit Grover
Assistant Professor (ECE),
SBSSTC, Ferozepur

ABSTRACT
The sensor networks consist of the smaller sensor nodes which automatically constructs the routes between themselves to form the data delivery paths.There are several methods of energy conservation such as clustering, routing, duty cycling and many others. The duty cycling or sleep wake cycling is incorporated for the minimization for energy consumption. The effective duty cycling procedure requires the optimal selection of the sleep wake pairs adaptively constructed between the nodes, out of which one node goes sleep and another stays wake a particular point of time. In this paper, the adaptive optimal pairing with flexible routing has been utilized for the purpose of duty cycling in the sensor nodes. The proposed model has undergone several experiments and performed to be efficient enough. The proposed model has outperformed the previous models for duty cycling.

Keywords
Sleep wake scheduling, duty cycling, smart routing, and flexible routing across pairs.

1. INTRODUCTION
By using on-demand routing protocols, routes are discovered when a request is generated and then routing protocol obtain the possible routes 'as needed' [1]. In DUTY CYCLING, route discovery works as follows the paired nodes. Whenever a traffic source needs a route to a destination from the paired nodes for the duty cycling, it initiates a route discovery by flooding a route request (RREQ) for the destination in the network after that waits for a route reply (RREP)[2] [3]. When any intermediate node receives the RREQ packet, it sets up a reverse path to the source by using the previous hop to the RREQ as the next hop on the reverse path. In addition, if there's a route available to the destination, it unicasts a RREP message back to the source node via the reverse path; else, it again broadcasts the RREQ packet. RREQ's Duplicate copies are immediately discarded [4]. The destination node on receiving the first copy of a RREQ message forms a reverse path same as the intermediate nodes; it also unicasts a RREP message back to the source with the reverse path. As RREP packet proceeds to the source node, it establishes a forward path towards the destination. Route maintenance is done by means of route error (RERR) packets. If there any node failure detection at intermediate node (via a link-layer feedback, e.g.), it will generate a RERR packet. The RERR erases all broken routes on the way. A source node, receiving the RERR initiates a new route path if it still wants it. Apart from this route maintenance mechanism, DUTY CYCLING also has a timer-based mechanism to purge stale routes [5].

Sequence numbers in DUTY CYCLING play a key role in guaranteeing loop freedom. Every node maintains a monotonically increasing sequence variety for itself. It also maintains the highest familiar sequence variety for every destination within the routing table known as the 'destination sequence variety.' Destination sequence numbers are labeled on all routing messages. They are used to confirm the relative freshness of two items of routing data generated by two nodes for identical destination—the node with a better destination sequence variety has the more modern routing data [6]. The DUTY CYCLING protocol prevents routing loops by maintaining an invariant that a sequence number for destination along with any valid route increasing monotonically towards the destination. This idea is again elaborated below because ADPATIVE ROUTING also uses a similar invariant for loop freedom [7].

Our objective in this section is to extend the DUTY CYCLING protocol to compute multiple disjoint loop-free way's in a route discovery. We assume that each node includes a unique symbol (UID) (e.g., IP address). For simplicity, authors also assume that all links are two-way, that is, a link exists between a node i to j if and only if there's a link from j to i. ADPATIVE ROUTING also applied in the presence of unidirectional links with additional techniques to help discover bidirectional paths in such scenarios [8].

The model can represent all the possible sensor schedule choices by a tree structure. Nodes depth in a tree represents time instants with root node at time zero. The branches correspond to choosing a specific sensor to move at that time instant. Thus, the path from root node to any node at depth d represents a specific sensor schedule alternative for time steps zero to d [9]. This model will associate with each node the value function evaluated using the sensor schedule similar to the path from the root node to that node. Obviously, finding the optimal sequence needs traversing all the ways from the root to the leaves. This procedure might place too high a demand on the procedure and systems memory. Hence authors would like some kind of on-line optimization procedure [10]. We present some approximations that address these difficulties. The first 2 approximations aim at pruning the tree thus on keep it to a manageable size whereas attempting not to lose the optimal sequence. The objective of third algorithm is to traverse the tree altogether though it minimize solely the expected steady state error variance. Currently these following 3 schemes have been considered.

Algorithm 1: Sliding window: it is similar to the Viterbi algorithm's pseudo real time version. In this, we define a window size d where d < N. The steps for successful execution of algorithm are defined as following:

1. *Initiate root node with time k = 0.*
2. *(A) Traverse tree with all the possible paths from current node to next upcoming nodes.*

   (b) Identify the sequence of sensors Sk,Sk+1,Sk+2,. . . , Sk+d−1 which at the end produce the minimum cost of this window size d.

   (c) From the sequence choose the first sensor Sk.

3. *(A) If time k = N then quit otherwise jump to the next step.*
    a) *Assign sensor Sk as the root node.*
    b) *Increment time as k = k + 1.*
    c) *Repeat traversal step 2 till the end.*

The window size d is an absolute parameter. If it is large enough, the sequence yielding the lowest cost can correspond the optimum sequence for the whole time horizon. Also note that once author slide the window, we already have the error covariances for the primary d − one time steps stored; therefore they do not have to be recalculated.

Algorithm 2: Thresholding: This algorithmic program is similar to that proposed in [7], in the context of selecting the optimal controller from a collection alternatives. We outline a cut-off factor f ≥ 1. The algorithm proceeds as follows:

1. *with cost 0, Start from root node*
2. *(a) from the current node traverse the tree by one step through all possible paths.*
    b) *Compute the minimum cost till that time step.*
    b) *Prune away any branch that yields cost larger than the minimum of f times*
    c) *For the remaining branches, denote the nodes cost as the cost gained by moving down the tree till the leaf node.*
3. *Every node considered as the root and repeats the pruning step 2 until reaches to the end.*
4. *After a sufficiently large time or N time steps interval, declare the optimal sequence to be the one yielding the minimum cost till that time step.*

The main objective to design the algorithm is to check the sequence whose value is too high at any intermediate time and that would not be producing the minimum cost. By playing with the factor f, we get a trade-off between the certainty that we'd not prune away the optimum sequence and therefore the range of branches of the error variance tree that require to be traversed.

Algorithm 3: arbitrarily Chosen Sensors: In this technique, at each step, the sensors are chosen at random according to some likelihood distribution, such that the i^{th} sensor is chosen with likelihood q^i. The probability distribution is then chosen thus as to minimize the expected steady state error variance. Note that we can't calculate the actual value of the error variance since that may rely on the specific sensing element schedule chosen. Hence author optimizes the expected steady-state value of the error variance. Thus authors are interested in

$$E[P[k+1]] = E\left[BQB^T + AP[k]A^T\right] - \Delta,$$

where Δ equals

$$E\left[AP[k]C[k]^T(R[k] + C[k]P[k]C[k]^T)^{-1}C[k]P[k]A^T\right].$$

The quantity R[k] is that the sensor noise that depends on the actual sensor chosen at time step k. explicitly evaluating this expectation seems to be unmanageable. And may get an upper bound as follows [8]. Firstly the quantities C[k] and P[k] are not dependent on anything. Where P is positive semi-definite and R is positive definite, APCT (R + CPCT) −1CPAT is convex in P and can apply Jensen's inequality. Using these facts yields the upper bound.

$$E[P[k+1]] \leq BQB^T + AE[P[k]]A^T - \sum q_i\Delta_i, \quad (3)$$

where Δ_i equals

$$A\left[E[P[k]]C_i^T(R_i + C_iE[P[k]]C_i^T)^{-1}C_iE[P[k]]\right]A^T.$$

Following [9], It can prove that as long as A is stable, the changed Riccati recursion in equation (3) converges and the mean of error variance is that the distinctive positive semi-definite resolution of the corresponding modified algebraically Riccati equation. Note that a being stable is the usual case in practical applications of estimation. The algorithm therefore consists of selecting qi's thus on optimize the boundary as a method of optimizing the expected steady state value of Pk itself. The drawback of optimization is to resolve the underneath constraint that all the qi's should be non-negative and will add up to one. The problem will be resolved by the gradient search algorithm or maybe by brute force search for small N.

2. EXPERIMENTAL DESIGN

The proposed system can be improved and increased for its metric calculation to elect the simplest route and route for load equalization whereas causing the information towards the BTS. The BTS will be receiving the information from the cluster heads within the wireless networks. The metric calculation would be improved by combining the values of the next hop energy, all hop energy, hop count, nodes id and bandwidth between the source node and destination node. The adaptive load balance equalization rainbow protocol can use this new metric route to realize the shortest route with balanced energy and better information measure. The route cost calculation for load equalization can be supported the individual load on the relay node/s, various choice of routes with the minimum load will be also thought of to search out the alternative route. Among the shortlisted routes using the load as metric, the route with minimum total route cost can be used to forward the information. The proposed model can be compared with the existing system using end to end delay/latency, packet delivery ratio, network/route load and packet Efficiency of the different route. The project will be developed by using the NS2 simulator. The algorithm to find the connectivity hole or link failure can be used to update the routing table whereas the first route becomes inaccessible. This method will be using reply back methodology to find the link failure, and to execute the backup and load balance route finder event based improved adjustive load balancing rainbow protocol for WSNs.

Property	**Value**
Transmission Radius	250 m
Initial Coordinates	Random
Queue Length	50 packets
Wireless Channel	Standard Wi-Fi
Link Layer Type	DLL
Routing Protocol	AODV
Number of Nodes	100

Connectivity Model	MAC 802.11
Antenna Model	Omni Antenna
Simulation Area	1500 x 1000

Algorithm 4: Weighted Optimal Duty Cycling

1) Node N starts up.
2) Nodes N starts negotiating with the nodes within transmission reach of every node i.
3) Node i builds its neighbor table with initial cost parameter based upon distance, bandwidth & neighbor node id.
   $d(pi) = ((dx_i - dx_j) + (dy_i - dy_j))^2$ ---------- (1)
   $bw(pi) = \max(e1, e2, e3 \dots \dots \dots en)$ ---------- (2)
   $nid(pi) = \max(bestPath(nid1, nid2 \dots \dots \dots nidn)$ ---------- (3)

$P_P(i) = \int_1^N P(E_i) < R^2$, using (1), (2) and (3)

   Where, $P(E_i)$ denotes distance and R denotes Radius, $P_P(i)$ denotes path distance, $P(E_i)$ is Path Energy $P(D_i)$is Distance.

4) A query 'a' sends by the Node i to the neighbor node to find the path to the sink node.
5) If the neighbor node NN_i knows the path to the sink,
   a) it will reply with route information to the querying node i.
   b) node i will update the routing information
6) Else/Otherwise,
   a) Neighbor node NN_i will pass the query to its querying node & it will continue till it reaches the sink node.
   b) Once the path to sink is found, nodes will send the route reply towards the querying node.
7) Find the optimal neighbor node with minimum distance.
8) Perform the node pairing between the two nodes
9) Find the best optimal path for both of the paired nodes for the duty cycling in the sensor network
10) Repairing of the node occurs on the points of no route to build the successful end-to-end routes
11) The path with lower cost is found and selected.
12) Data is channelized through the selected path.

The path forwarding is the process of the selecting the path to send the data and then sending the data over the selected path. For the path selection there are various parameters, which are evaluated to compute the runtime path cost in order to find the best available path towards the destination. The priority wise parameters indexing can be defined as: energy, bandwidth and neighbor node id (node at distance of one-hop). The lowest energy node is preferred from the path energy matrix, whereas the highest bandwidth and neighbor node id has been used for the tie-breaking when the energy will be equal across multiple paths.

3. RESULT ANAYLSIS

Throughput: Throughput is the parameter to define the rate of successful data delivered on the received end. The throughput depicts capacity of the network to deliver the certain amount of data in the given time slot.

The throughput analysis has been produced using the two stages:

1. With fixed trust factor, and
2. With fixed time factor.

Trust Factor: The trust factor is the factor of the trust between the two sensors in the sensors in the given pair. The trust defines the compatibility between the two sensors. The trust factor between 0.8 and 1.0 is considered the best level in the communication channel.

Time Factor: The time factor is the factor used between the two devices, which schedule the sensors for the given span of time.

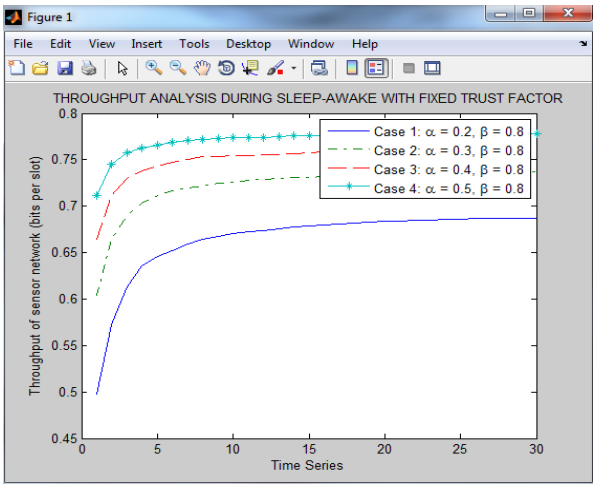

Figure 1: Throughput analysis during sleep-awake with fixed trust factor

The figure 1 defines the four cases with the fixed trust factor on 0.8. The time factor is being increased gradually from 0.2 to 0.5. The trends have shown the significant improvement in the case of throughput while the time factor is being increased.

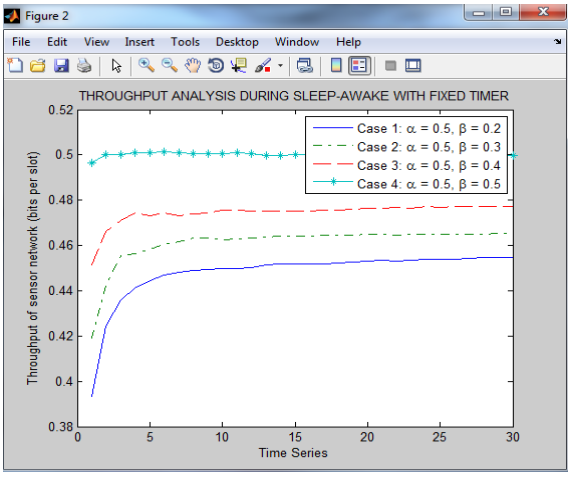

Figure 2: Throughput analysis during sleep-awake with fixed time factor

The figure 2 defines the four cases with the fixed time factor on 0.5. The trust factor is being increased gradually from 0.2 to 0.5. The trends have shown the significant improvement in the case of throughput while the trust factor is being increased.

WSN Lifetime Analysis

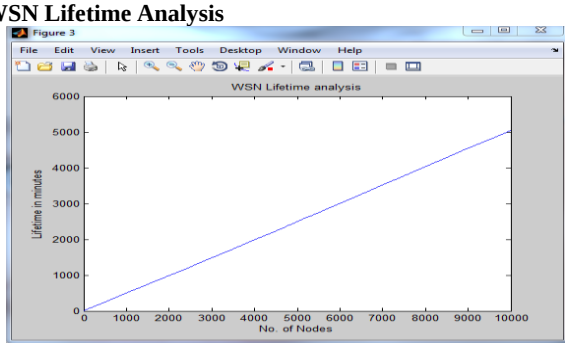

Figure 3: Lifetime analysis of the wireless sensor network

The lifetime of the wireless sensor network has been analyzed in the case of varying number of nodes. With the rise in the number of nodes, the lifetime of the WSN has been shown improving in the straight curve.

Access Behavior of Mobile Users

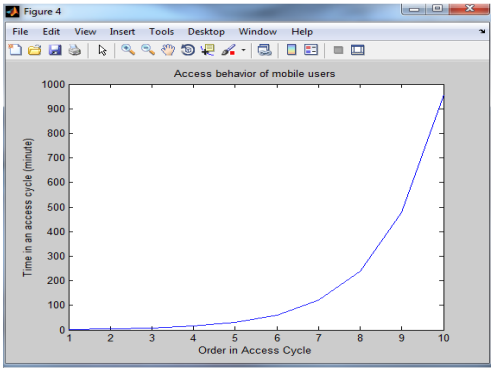

Figure 4: Access behavior of mobile users

The access behavior of the mobile users has been analyzed in the given WSN topology using the various orders in the access cycle. The access behavior of mobile users defines the time of access for per user. Every user is assigned the communication slot in the given network, in the different rounds or order. The time of access cycle for every user rises with each round. The network becomes well converged with the passage of rounds.

Energy Based Analysis

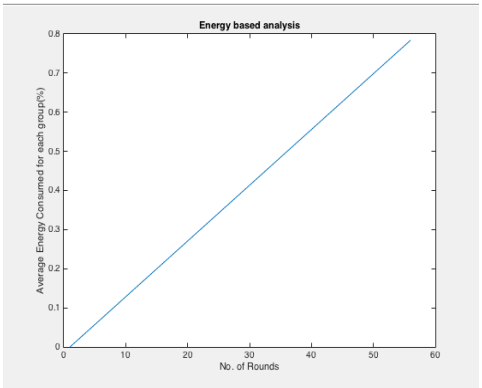

Figure 5: Energy Based Analysis

The Energy is the parameter represents the consumed energy of a network in entire transmission. The sensor network using our proposed energy aware sleep scheduling based routing protocol has been recorded between 0and 0.8 joules in proposed system and 0 to 0.9 joules in the existing system. The 0 joules is the value recorded when no data is being sent between the nodes in the initial stages. Once the data transfers start, the energy starts going down.

Latency Based Analysis

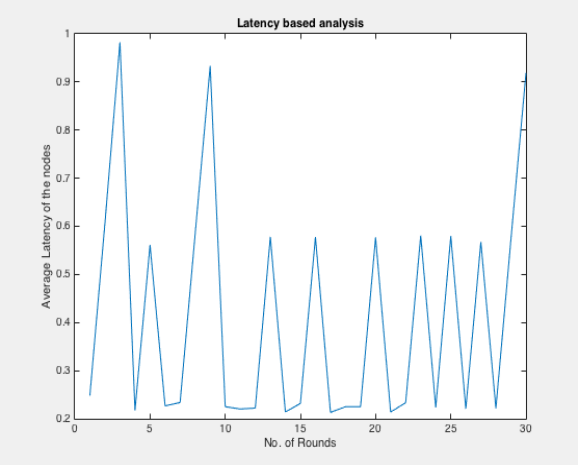

Figure 6: Latency Based Analysis

The maximum delay recorded in the simulation is ranging between 0.25 and 0.95 milliseconds. (Figure 7) The delay is the parameter represents latency of a packet when it was being sent between two nodes. The time taken for a packet to reach the destination from the source is called the total delay.

4. CONCLUSION

The duty cycling approach has been proposed for the higher order of energy efficiency among the wireless sensor networks. This approach is based upon the adaptive pair generation mechanism to bind two sensor nodes altogether for the sleep wake scheduling. When first node goes operational, the other went sleeping and vice-versa. The proposed model has been tested with various numbers of nodes and topological parameters and the performance has been found consistently adaptive and stronger under all situations. The proposed model has outperformed the existing model in energy efficiency and throughput.

5. FUTURE SCOPE

The multiple group duty cycling approach can also be utilized to reduce the parametric effort from the bi-group paradox. Also the multi-function WSN deployment can be considered for the hierarchical consideration with effort minimization based aggregation over the meta-heuristic traffic generated from the heterogeneous and meta-heuristic WSN architecture.

6. REFERENCES

[1] Zhu, Chunsheng, Victor Leung, Laurence T. Yang, and Lei Shu. "Collaborative location-based sleep scheduling for wireless sensor networks integrated with mobile cloud computing." (2014).

[2] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, “A survey of mobile cloud computing: Architecture, applications, and approaches,” Wireless Commun. Mobile Comput. vol. 13, no. 18, pp. 1587–1611, Dec. 2013.

[3] S. Wang and S. Dey, "Adaptive mobile cloud computing to enable rich mobile multimedia applications," IEEE Trans. Multimedia, vol. 15, no. 4, pp. 870–883, Jun. 2013.

[4] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," Future Generation Comput. Syst., vol. 25, no. 6, pp. 599–616, Jun. 2009.

[5] C. Zhu, L. Shu, T. Hara, L. Wang, S. Nishio, and L. T. Yang, "A survey on communication and data management issues in mobile sensor networks," Wirel. Commun. Mob. Comput., vol. 14, no. 1, pp. 19–36, Jan. 2014.

[6] M. Li and Y. Liu, ".Underground coal mine monitoring with wireless sensor networks," ACM Trans. Sens. Netw, vol. 5, no. 2, Mar. 2009

[7] M. Yuriyama and T. Kushida, "Sensor-cloud infrastructure - physical sensor management with virtualized sensors on cloud computing," in Proc. 13th Int. Conf. Netw.-Based Inf. Sys. (NBiS), 2010, pp. 1–8.

[8] G. Fortino, M. Pathan, and G. D. Fatta, "Bodycloud: Integration of cloud computing and body sensor networks," in Proc. IEEE 4th Int. Conf. Cloud Comput. Technol. Sci. (CloudCom), 2012, pp. 851–856.

[9] R. Hummen, M. Henze, D. Catrein, and K. Wehrle, "A cloud design for user-controlled storage and processing of sensor data," in Proc. IEEE 4th Int. Conf. Cloud Comput. Technol. Sci. (CloudCom), 2012, pp. 232–240.

[10] Y. Takabe, K. Matsumoto, M. Yamagiwa, and M. Uehara, "Proposed sensor network for living environments using cloud computing," in Proc. 15th Int. Conf. Netw.-Based Inf. Sys. (NBiS), 2012, pp. 838–843.