Roy Bean Algorithm: Distributed Load Balancing for Fairness-Efficiency Enhancement in Wireless Sensor Networks

Xu Li [*] Liang Cheng [†] Huan Yang [‡]

Abstract

In wireless sensor networks, limited battery power imposes serious constraints on the lifetime of individual sensor nodes. On the other hand, each sensor node's efficient utilization of its own battery power does not necessarily result in network-wide fair distribution of energy or improved network lifetime. This gap between local efficiency and global fairness has motivated the development of assorted load-balancing algorithms. In this report, we explore the extension of network lifetime through the balance between energy efficiency and fairness. We develop Roy Bean algorithm, a cross-layered and distributed load-balancing algorithm that is capable of achieving a flexible trade-off between efficiency and fairness. We also perform theoretical analysis and simulation that substantiate not only our distributed implementation of Roy Bean algorithm but any rescheduling operation that improves local fairness. We then refine our algorithm and resolve issues pertaining to hardware implementation. Our lab experiment demonstrates that Roy Bean algorithm improves the uniformity of sensor node lifetime and thus improves the serviceability of battery-powered WSNs in real applications.

1 Introduction

The popularization of wireless sensor networks (WSNs) and the continuous performance improvement of both microcontrollers and wireless transceivers have not only enabled large-scale deployment of interconnected sensors but also introduced technological challenges such as improving energy efficiency and extending network lifetime. Nowadays, increasing instances of WSN employ battery-powered sensor nodes which usually communicate in a multi-hop fashion. In addition to transmitting locally generated packets, sensor nodes also need to relay packets for its neighbors. For individual sensor nodes, energy efficiency is typically defined as the ratio of energy consumed by transmitting locally produced packets to the total energy consumption by all tasks including sensing, data processing, as well as receiving and transmitting packets. Evidently, relaying operations on each sensor node should be minimized in order to achieve higher energy efficiency. Moreover, the definition of energy efficiency for individual nodes also suggests that each node be better off staying "selfish", which, in extreme cases, leads to network partitioning since every node refuses to serve other nodes.

[*]Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA, USA. E-mail: `lixulehigh@gmail.com`

[†]Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA, USA. E-mail: `cheng@cse.lehigh.edu`

[‡]Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA, USA. E-mail: `huy213@lehigh.edu`

Fairness, on the other hand, is a network-wide metric assessing the degree to which workload and energy are equally distributed. In WSNs, fairness is commonly implemented through load balancing as each node's battery normally has the same capacity. The fundamental idea behind load balancing algorithms is that nodes with relatively more residual power should forward packets for those with less remaining power. Therefore, load balancing approaches regard the total battery power of a WSN as a resource and attempt to allocate it as fair as possible. In other words, load balancing algorithms generally promote fair use of network energy and relieve congestion, resulting in a prolonged network lifetime.

In real world applications such as structural health monitoring (SHM) WSNs, we observe that the balance between efficiency and fairness determines the serviceability of the network. In [5], the authors deploy sensors along a bridge to measure its modal shapes. Each sensor provides a sampling point on the bridge and most sensors not only need to transmit their own packets but have to relay other nodes' packets as well. In order to obtain an accurate modal shape, each sensor must sample enough data through its accelerometer and transmit these data to the data sink; meanwhile, the deployed WSN must support the collection of acceleration data from as many sensor nodes as possible. In other words, the design specification of such WSNs requires both efficient use of energy when transmitting locally gathered data as well as cooperation between sensor nodes to achieve fair and effective delivery of packets originated from other nodes. Both fairness and efficiency requirements are essential to the accurate calculation of modal shapes. In fact, there have been considerable discussions of combining multiple objectives into a single optimization model, either by including certain objectives as constraints or by assigning weights to different objectives. However, Zhao [14] has shown that neither of those techniques can effectively achieve a fair trade-off between efficiency and fairness.

In this report, we present Roy Bean, a novel load-balancing algorithm that achieves a better and more flexible balance between fairness and efficiency. We provide mathematical basis for our proposed distributed, cross-layered load-balancing operation, which also substantiates future design and implementations of distributed variants of existing load-balancing algorithms. In addition, we set up simulation to evaluate the performance of Roy Bean algorithm and carry out comparisons between Roy Bean and other well-accepted algorithms under various scenarios. We also implement and refine Roy Bean algorithm in lab experiments, which corroborates our theoretical analysis. The rest of the report is organized as follows: We first survey state of the art load-balancing algorithms and research on fairness indices. Then, we formulate the problem we are interested in, present our own approach toward a better balance between energy efficiency and fairness, and design the Roy Bean algorithm. Next, we demonstrate through simulation that Roy Bean algorithm outperforms other well-known load-balancing algorithms. Finally, we implement and refine Roy Bean algorithm on Imote2 smart sensor platform.

2 Related Work

An initial motivation of several load-balancing algorithms, such as Popa [12] and Pathak [11], is the desire to avoid premature depletion of battery power of nodes located on congested routes. Observing these "hot spots", Popa [12] maps the deployment plane onto a sphere so that packets can also flow through adjacent paths, which have the same cost as that of the shortest path when mapped onto a sphere. As a result, traffic flow over the WSN is more evenly distributed. Since the shortest path on the deployment plane is still one of the shortest paths on the sphere, it is easy for routing protocols based on distance vectors to adopt similar mapping schemes. Nevertheless, the mapping process requires extra computations and the locations of the nodes not only on the deployment plane but also on the routing sphere, which introduces considerable complexity.

In contrast to the routing-based solution proposed by Popa [12], Pathak [11] addresses the same issue through transmission power control. Assuming randomized traffic patterns, Pathak [11] observe that congestion tends to occur at the geometrical center of the deployment plane. Correspondingly, adjusting transmission power according to the centrality [11] of a node, which is defined as a quantified score of its expected traffic load, results in improved load distribution and extended network lifetime. Despite achieving fairer use of energy, the cross-layered solution by Pathak [11] demands the calculation of each node's centrality score, which in turn depends on global knowledge of network topology and limits the scalability of the WSN. Furthermore, both Popa [12] and Pathak [11] focus on WSNs with randomly scattered nodes and randomized traffic patterns. In the aforementioned SHM example [5], we observe that actual traffic patterns are less random and certain sensor nodes always need the assistance from other relaying nodes to complete data transmission. We will further illustrate this issue in our simulation and demonstrate how Roy Bean algorithm avoids this problem.

To measure fairness performance of load balancing algorithms, researchers evaluate different algorithms in terms of network lifetime. A commonly-used network lifetime definition, adopted by Giang [6], Chang [3] and Hsiao [7], is the time span between network initialization and the failure of the first node. In many applications such as [5], failure of an individual node does not necessarily signify the failure of the entire WSN. In fact, WSNs with densely deployed sensor nodes will only become partitioned when a significant amount of nodes stop working. In such scenarios, performance metric based on shortest lifetime cannot well represent fairness over the whole WSN. In our simulation, we monitor the lifetime of every individual node and plot the data to illustrate the fairness performance of Roy Bean and comparing algorithms.

In addition to network lifetime, alternative fairness indices have been studied in miscellaneous disciplines of applied science and economics. As classified by Tian [9], many frequently used fairness indices, including max-min index and Jain's index [8], fall into the category of scale-invariant fairness measures [9]. The first-node-to-die definition of network lifetime essentially employs max-min index, which focuses on fair allocation of an arbitrary portion of available resources. Consequentially, WSNs that quickly exhaust large-capacity battery of a single node can be as fair as those evenly draining small-capacity batteries of all nodes. Jain's index [8] has the same issue: As long as an equal distribution of certain resources is achieved, the overall fairness index will be 1. In order to design a load-balancing strategy that takes into account not only fairness among sensor nodes but also their respective energy efficiency, we adopt the fairness measure proposed in [9] and extend its use in WSNs to facilitate distributed algorithm design..

3 Problem Formulation

3.1 Network Configuration

In order to devise a load balancing strategy for practical WSNs with relatively large scale and non-random communication patterns, we assume a WSN topology similar to that of [5]: A single data sink collects data from all the sensor nodes deployed across a bridge. During every cycle, all sensor nodes generate their own packets containing sensing data. Nodes within one-hop distance of the sink can directly transmit their own data packets without interaction with any relay nodes, while other nodes have to resort to intermediate nodes. Both our simulation and field experiments closely follow this commonly-seen network configuration. The performance of existing algorithms, most of which assume the classic random graph model [2], are also evaluated under the same configuration.

3.2 Design Objective

The primary objective of our algorithm is to achieve a controllable trade-off between energy efficiency of individual sensor nodes and fairness in terms of traffic forwarding. We adopt the fairness index proposed in [9], which takes the form of

$$F(\mathbf{x}) = f(\mathbf{x}) \cdot |\mathbf{x}|^{\lambda} \tag{3.1}$$

where $f(\mathbf{x})$ is a fairness measure. An example of $f(\mathbf{x})$ is the Jain's index [8] in the form of

$$f(\mathbf{x}) = \frac{|\mathbf{x}|^2}{n\Sigma_{i=1}^{n} x_i^2} \tag{3.2}$$

As mentioned in Section 2, most fairness indices fail to take into account whether all available resources are allocated and may sometimes lead to a fair but poor allocation scheme. Therefore, our definition in Eq. (3.1) combines a fairness measure with the resource allocation vector $|\mathbf{x}|$ and the preference parameter λ. The unit used in resource allocation vector $|\mathbf{x}|$ is bits-per-Joule which is proposed in [13]. The definition in Eq. (3.1) can evaluate the performance of two resource allocation schemes that utilize the same portion of overall allocable resources, but it is not capable of comparing schemes using different fractions of the overall resources. To remove this limitation, we normalize $|\mathbf{x}|^{\lambda}$ using the overall available resources and define an alternative metric as follows

$$F(\mathbf{x}) = f(\mathbf{x}) \cdot \frac{|\mathbf{x}|^{\lambda}}{\Sigma_{i=1}^{n} |x_i|^{\lambda}} \tag{3.3}$$

where $|x_i|^{\lambda}$ denotes the total accessible resources on each individual sensor node i. In our simulation and experiments, we monitor the WSN until the exhaustion of batteries of all the sensor nodes. Hence, both Eq. (3.1) and Eq. (3.3) can be regarded as equivalent metrics of the balance between energy efficiency and fairness. We use Eq. (3.1) in all ensuing sections of this report.

4 Algorithm Design

4.1 Theoretical Analysis

Before introducing fairness indices into the design of load-balancing algorithms, we have to point out a key distinction between redistributing resources and rescheduling WSN links. Redistribution will not cause any loss of resources. When comparing two different resource allocation schemes, we can consider one scheme as the redistribution after the other scheme's implementation. Rescheduling of WSN links, on the other hand, will inevitably introduce communication overhead and possible retransmission of packets, which is the cost of achieving a better balance between energy efficiency and fairness. Hence, a load-balancing algorithm should also take into consideration the loss of resources due to rescheduling. To emphasize the difference between the ideal concept of redistribution and the realistic rescheduling operation in WSNs, we define a reallocation that will incur loss of overall resources as a Roy Bean Operation.

Variable	Meaning
$\mathbf{x}$	resource allocation vector
x_i	resource allocated to node i
$\bar{x_i}$	resource allocated to nodes other than i
$g(\cdot)$	generator function
$h(\cdot)$	mean function
s_i	possitive weight for weighted mean

Table 4.1: List of Notations Used in Theorem 4.2.

Definition 4.1. *A Roy Bean operation is the lossy redistribution of overall available resources that takes abundant resources from some individual members and allocates them to those in need.*

As the key operation in our proposed algorithm, a Roy Bean operation reschedules the packet-forwarding links and relieves energy-starved nodes of extra burden of relaying. Each node monitors its residual battery power and negotiates with neighboring nodes about handing over its relaying liability. To demonstrate the correctness of Roy Bean algorithm, we need to prove that our Roy Bean operation satisfies the axiom of homogeneity exactly as an ideal redistributing operation does. The notations used in the following theorems are listed in Table 4.1.

Theorem 4.2. *Given a resource distribution $\boldsymbol{x}$, for any pair of elements x_i and x_j, if there is a new distribution $\{x_i', x_j'\}$ satisfying both $F(x_i', x_j') \geq F(x_i, x_j)$ and $x_i' + x_j' \leq x_i + x_j$), then $F(x_i', x_j', \bar{x_{ij}}) \geq F(x_i, x_j, \bar{x_{ij}})$*

Proof. As discussed in [9], a fairness index conforms to the "axiom of partition", thus

$$\frac{f(x_i, x_j, \bar{x_{ij}})}{f(x_i', x_j', \bar{x_{ij}})} = h\left(\frac{f(x_i, x_j)}{f(x_i', x_j')}, \frac{f(\bar{x_{ij}})}{f(\bar{x_{ij}})}\right) = g^{-1}\left[s_1 \cdot g\left(\frac{f(x_i, x_j)}{f(x_i', x_j')}\right) + s_2 \cdot g\left(\frac{f(\bar{x_{ij}})}{f(\bar{x_{ij}})}\right)\right] \tag{4.1}$$

Where x_i' and x_j' is the new distribution between node i and j. As Equation (3.1) suggests, we choose $g(\cdot)$ as a power function. Specifically, $g(x) = x^{\frac{1}{\lambda}}$. Also, we choose the weight s_1 and s_2 to be proportional to the resources allocated to $\{x_i, x_j\}$ and $\bar{x_i}j$, respectively.

$$s_1 = \frac{x_i + x_j}{x_i + x_j + \bar{x_{ij}}} \tag{4.2}$$

$$s_2 = \frac{\bar{x_{ij}}}{x_i + x_j + \bar{x_{ij}}} \tag{4.3}$$

Since the new distribution $\{x_i', x_j'\}$ is achieved through rescheduling such as Roy Bean operation, it follows from Eq. (3.1) and the two conditions satisfied by a new distribution that

$$f(x_i, x_j)(x_i + x_j)^\lambda \leq f(x_i', x_j')(x_i' + x_j')^\lambda \Rightarrow \frac{f(x_i, x_j)}{f(x_i', x_j')} \leq \left(\frac{x_i' + x_j'}{x_i + x_j}\right)^\lambda \tag{4.4}$$

Since $g\left(\frac{f(\bar{x_{ij}})}{f(\bar{x_{ij}})}\right) = g(1) = 1$, it follows from Eq. (4.1) and Eq. (4.4) that

$$g^{-1}\left[s_1 \cdot g\left(\frac{f(x_i, x_j)}{f(x_i', x_j')}\right) + s_2 \cdot g\left(\frac{f(\bar{x_{ij}})}{f(\bar{x_{ij}})}\right)\right] \leq \left(\frac{x_i' + x_j' + \bar{x_{ij}}}{x_i + x_j + \bar{x_{ij}}}\right)^\lambda \tag{4.5}$$

Therefore, $F(x_i', x_j', \bar{x_{ij}}) \geq F(x_i, x_j, \bar{x_{ij}})$. □

Theorem 4.2 suggests that any local improvement in the balance between energy efficiency and fairness lead to a global improvement in spite of extra rescheduling cost. In other words, Roy Bean operation can not only achieve an improved trade-off between efficiency and fairness but also accomplish such an enhancement in a distributed fashion. Furthermore, Theorem 4.2 also enables us to implement distributed versions of existing load-balancing algorithms.

4.2 Roy Bean Algorithm

We implement Roy Bean algorithm as a cross-layered load-balancing algorithm. Each sensor node keeps a table of its neighboring nodes; one of them is marked as the relay node, while others are registered in the table either as child nodes or as neighbors whose packets can be observed. In our algorithm, we denote the relay node of a sensor node as its parent node. A child node is a sensor node that needs the relaying service of its parent node to complete data transmission. Neighbors are reachable nodes that are neither parent nodes nor child nodes of a sensor node. We use a timer to trigger updating operation of this table. Each sensor node is capable of adjusting its transmission power and monitoring its battery voltage. Initially, each node broadcasts its node ID and table of neighboring nodes so that other nodes can collect necessary information to select its relay node.

When a relay node's battery voltage drops below a predefined threshold, it notifies all the neighbors through broadcasting both its request and current table of neighboring nodes. This relay node then starts a timer and waits for the responses from its child nodes. On the other hand, all the child nodes will attempt to find a new relay node that is one of either its own neighbors or its parent node's neighbors. If a new relay node takes over the responsibility, the child node will notify its original parent node. If the timer expires and some child nodes have not found any other relay node, the original relay node will still serve its remaining child nodes. When registering a new child node, the potential parent node has to check its table of neighboring nodes to prevent invalid routes such as cycles.

Our Roy Bean algorithm is a Bellman-ford-like algorithm with a similar time complexity of $O(VE)$, where V is the total number of nodes and E is the total number of communication links. However, Theorem 4.2 enables us to implement a distributed version of Roy Bean algorithm and take advantage of parallelism. More importantly, Theorem 4.2 also indicates that we are able to achieve improvement before the algorithm's ultimate convergence. In our simulation and experiments, we observe that a few local Roy Bean operations executed by nodes on critical routes significantly increase energy efficiency of individual nodes and prolong network lifetime.

5 Evaluation and Comparison Using Simulation

We conduct two sets of simulation to evaluate and compare the performance of our proposed algorithm and other representative algorithms using the network configuration described in Section 3.1. We use the Wireless Matlab Simulator [1] which supports the simulation of wireless ad-hoc networks and provides a discrete event scheduler.

In the first set of simulation, we compare Roy Bean algorithm with three representative algorithms: AODV, single hop and centrality routing. AODV represents routing layer protocols that have no control of the behaviors of lower layers. In single hop routing, every node attempts to directly communicate with the data sink using the lowest possible transmission power. It exemplifies WSNs where every node is "selfish" and tries to conserve its own energy. Centrality routing [11] is

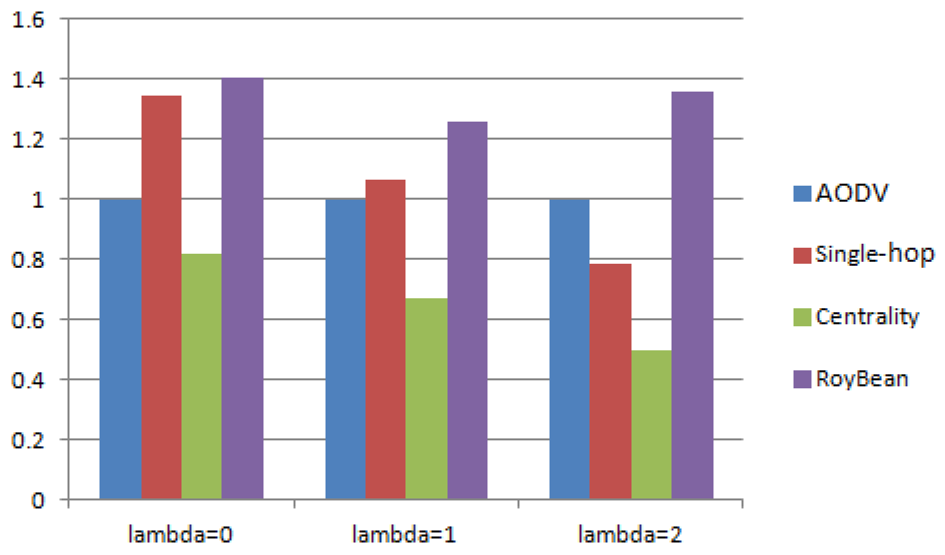


Figure 5.1: Performance Comparison through Simulation: Efficiency-fairness performance of algorithms is calculated using Eq. (3.1) and then normalized using the performance of AODV. Roy Bean algorithm outperforms other algorithms in terms of efficiency-fairness performance.

a multi-hop, cross-layered load-balancing algorithm that adjusts its transmission power according to each node's centrality score. On the other hand, our proposed algorithm is not only a multi-hop, cross-layered algorithm but also a distributed one that is capable of achieving a controllable fairness-efficiency balance.

We randomly deploy 100 sensor nodes on a $500 \times 500m^2$ plane with a data sink located at the center. Every node periodically generates a data packet and transmits it to the sink until the exhaustion of its battery power. Our simulation emulates the scenarios encountered in bridge sensing applications such as [5], where the nodes located near the sink inevitably serve as relay nodes for others. The parameter λ in Eq. (3.1) is set to 0 so that Roy Bean algorithm will focus solely on fairness.

The simulation results show that centrality routing has the worst efficiency-fairness performance due to the fact that traffic patterns in typical bridge sensing applications such as [5] are not randomized. Single-hop routing has a better performance than AODV and centrality routing because the batteries on all the sensor nodes are configured to have the same capacity. However, Roy Bean algorithm outperforms the others because it does not assume any particular traffic patterns and it allows localized improvement that overcomes the selfishness issue in single hop routing. According to the definition in Eq. (3.1), the efficiency-fairness performance of all the algorithms is depicted in Fig. 5.1 and all the results are normalized using the performance of AODV algorithm.

In the second set of simulation, we configure λ to be 1 and 2 for Roy Bean algorithm and carry out simulation using identical network configuration. The efficiency-fairness performance of all the algorithms is also included in Fig. 5.1. As mentioned in Section 3.2, our metric incorporates fairness with energy efficiency and our proposed algorithm is the only one that supports customizable trade-off between these two aspects. Consequently, Roy Bean algorithm exhibits better efficiency-fairness performance.

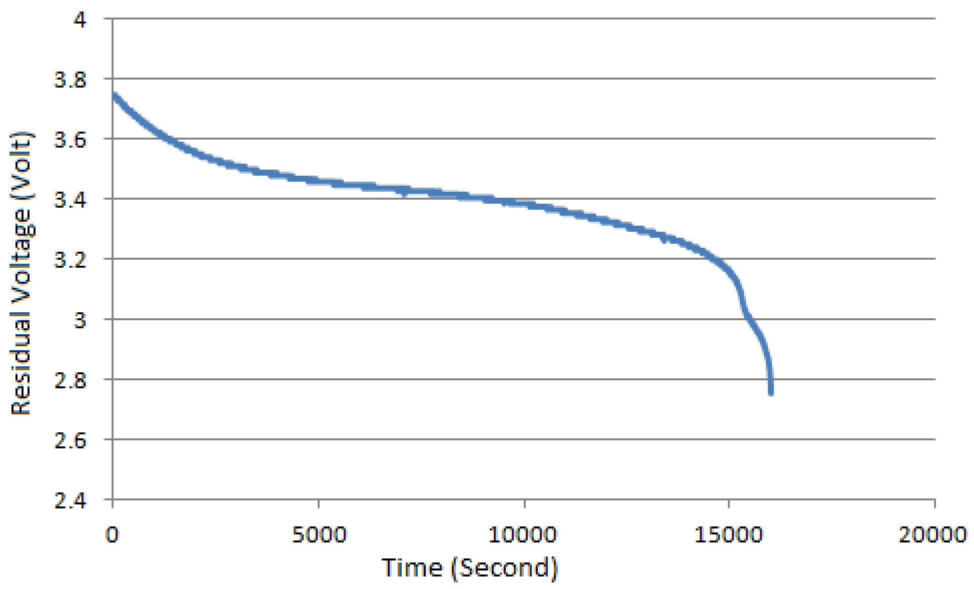


Figure 6.1: Battery Discharging Curve of a Sensor Node with 3 AAA Batteries.

6 Enhancements and Analysis through Experiments

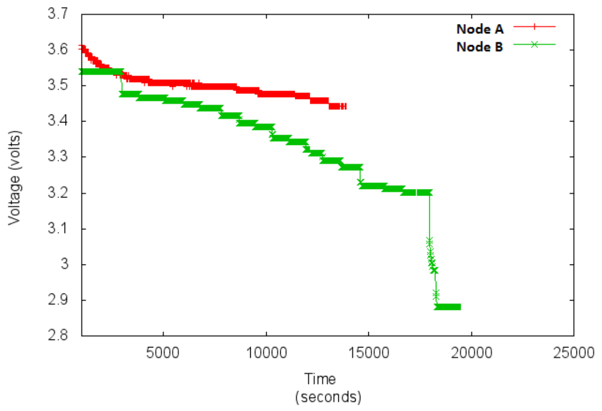
We implement Roy Bean algorithm on Imote2 platform running TinyOS [10], a lightweight open-source operating system for sensing devices. We first conduct small-scale, preliminary experiments to resolve issues pertaining to hardware implementations. Then, we proceed to implementation and evaluate our algorithm in realistic scenarios. We develop a benchmark application for sensor nodes. The benchmark program keeps sending time-stamped packets to the sink. The sink node will keep track of the latest time when it receives a packet from a certain node until the depletion of all batteries.

6.1 Preliminary Experiment

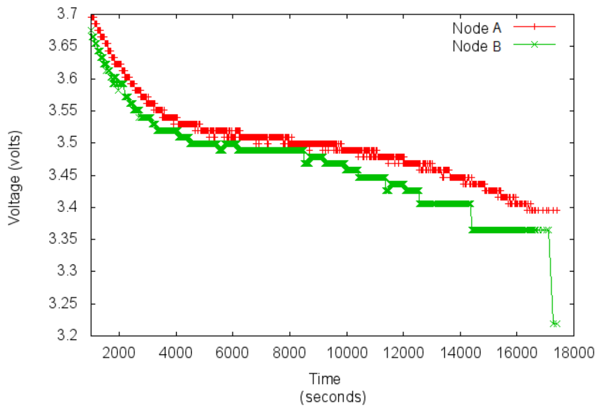
To implement Roy Bean algorithm on real sensor platforms, we must find an alternative way to measure residual battery power and estimate remaining lifetime of a sensor node. Therefore, we run our benchmark application on a single sensor node with 3 AAA batteries. The sensor node uses maximum transmission power available on Imote2. We plot the discharging curve of the sensor node on Fig. 6.1, where we can observe that the battery voltage stabilizes around 3.4 *volts* before a final steep drop to 2.8 *volts*. Since we cannot directly measure battery power, we choose 3.4 *volts* as the threshold voltage. Any sensor node with a battery voltage lower than this threshold is regarded as a candidate for Roy Bean operation.

We point out that this procedure is indispensable for hardware implementation and should be repeated if different type or number of batteries will be used in new implementation. Moreover, we also observed that sensor nodes will stop working well before its battery voltage drops to 0. As a result, we also need to refine our lifetime estimation model used in simulation.

Next, a small-scale linear network is implemented, with 2 sensor nodes neighboring each other and a data sink adjacent to one of the sensor nodes. The parameters of our benchmark program are



(a) Battery Discharging Curves of Two Sensor Nodes with $\lambda = 2$.



(b) Battery Discharging Curves of Two Sensor Nodes with $\lambda = 0.5$.

Figure 6.2: Performance of Roy Bean Algorithm in Small-scale Experiment.

listed in Table 6.1. The routing heartbeat is the frequency of core timer expiration which causes a sensor node to update its routing information. We monitor both the voltage levels of sensor nodes and their routing schemes. We first evaluate Roy Bean algorithm with $\lambda = 2$ and the results are shown in Fig. 6.2a. Node B is close to the sink and uses less transmission power to communicate with the sink. When λ is 2, Roy Bean algorithm favors energy efficiency and node B refuses to forward packets for node A, leading to the early depletion of the batteries on node A.

Parameter	Value
Transmission Rate	1 packet per 5 seconds
Routing Heartbeat	$1/9$Hz
Monitoring Frequency	$1/20$Hz

Table 6.1: Parameters Used in Small-Scale Experiment.

We then evaluate our algorithm with $\lambda = 0.5$. In this case, Roy Bean algorithm focuses more on fairness and the lifetime of node A is extended. The corresponding results are plotted on Fig. 6.2b. Although extra communication cost is induced by Roy Bean operation, we observe that the lifetime of both sensor nodes become uniform. In real-world applications such as [5], WSNs with uniform lifetime will provide more data with better quality. Furthermore, our experiment also demonstrates the flexibility of Roy Bean algorithm.

6.2 Refinement of Lifetime Prediction Model

Accurate estimation of sensor node lifetime is a critical task for Roy Bean algorithm. We observe that a sensor node stops working when its battery voltage drops below 2.8 *volts*, but the lifetime estimation model used in both our simulation and small-scale experiments does not address this issue. Instead, we use the following model

$$Lifetime = \frac{V_{residual}}{W \times P_{tx} \times \tau} \tag{6.1}$$

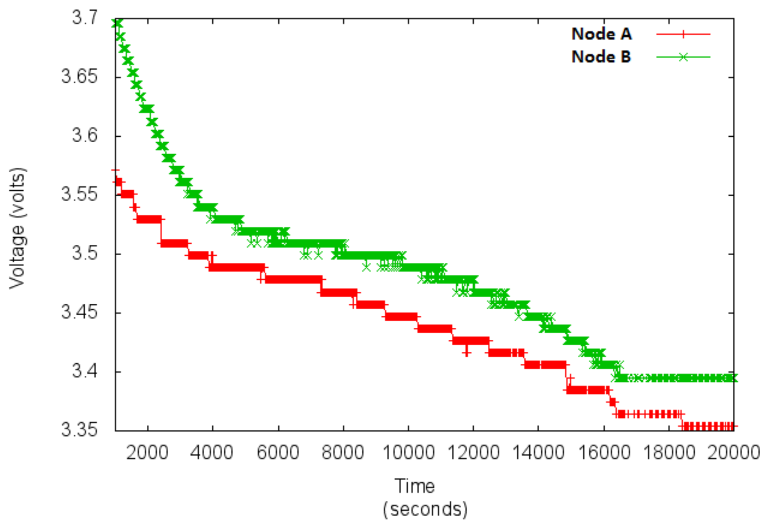


Figure 6.3: Battery Discharging Curves of Two Sensor Nodes with $\lambda = 0.5$ and $V_{floor} = 2.8\ volts$.

where $V_{residual}$ is an indirect measurement of residual battery power, P_{TX} is the transmission power for effective communication, and τ is a constant.

We introduce another threshold voltage to model the actual behavior of the batteries and Eq. (6.1) becomes

$$Lifetime = \frac{V_{residual} - V_{floor}}{W \times P_{tx} \times \tau} \tag{6.2}$$

where V_{floor} is 2.8 $volts$ in our Imote2 implementation. We modify our implementation and evaluate the performance of Roy Bean algorithm with $\lambda = 2$ and the corresponding results are shown in Fig. 6.3. Our results suggest that the integration of Eq. (6.2) into Roy Bean algorithm improve both network lifetime and its uniformity.

In addition to Eq. (6.2), we also explore the incorporation of a hard-coded voltage-to-lifetime conversion table. However, only marginal improvement is achieved because the non-linear portion of the discharging curve covers a relatively short time span. The hard-coded conversion table we considered is plotted in Fig. 6.4.

6.3 Lab Experiment Using Refined Roy Bean Algorithm

We implement our Roy Bean algorithm on an SHM damage detection test bed using 10 sensor nodes and a data sink. We install our benchmark application on all the sensor nodes and then turn on the sensor nodes simultaneously. We monitor the WSN until the depletion of all sensor nodes' batteries. The λ parameter of Roy Bean algorithm is 0.5. We observe that the actual lifetime of all sensor nodes are almost uniformly distributed: The maximum lifetime observed is $20,200$ seconds, while the minimum lifetime is $19,200$ seconds. This result indicates that Roy Bean algorithm does fulfill our design objective of a better and controllable balance between energy efficiency and fairness.

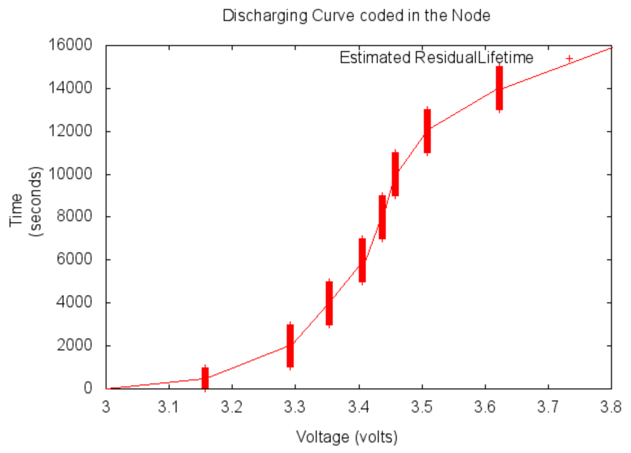


Figure 6.4: Voltage-to-lifetime Converstion Table Hard-coded in Sensor Nodes: Only marginal improvement is achieved. Therefore, this scheme is not used in lab experiments in the Section 6.3.

7 Conclusions and Future Work

In this report, we design a distributed load-balancing algorithm that achieves a controllable trade-off between energy efficiency and fairness. We propose a new efficiency-fairness metric and provide both theoretical and empirical evidence that substantiates the correctness, flexibility and superior performance of our proposed Roy Bean algorithm. In the future, we will explore other approaches to load balancing problems such as the game theoretic algorithm proposed in [4]. We will also consider investigating the scalability issues of various load balancing algorithms and explore the possibility of implementing distributed versions of existing load-balancing algorithms.

Acknowledgement

This material is based upon work supported by National Science Foundation under Grant No. 0915335. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Wireless Network Simulator in Matlab. Available Online at `http://wireless-matlab.sourceforge.net/`.

[2] Béla Bollobás. *Random graphs*, volume 73 of *Cambridge Studies in Advanced Mathematics*. Cambridge university press, 2001.

[3] Jae-Hwan Chang and L. Tassiulas. Maximum lifetime routing in wireless sensor networks. *Networking, IEEE/ACM Transactions on*, 12(4):609–619, 2004.

[4] Xiaoyu Chu and Harish Sethu. Cooperative topology control with adaptation for improved lifetime in wireless ad hoc networks. In *Proceedings of the IEEE INFOCOM*, March 2012.

[5] Siavash Dorvash, Shamim N. Pakzad, Rebecca C. Knorr, and Lauren M. Horwath. Modal Identification of Steel Truss Bridges Using Wireless Sensor Network. In *Proceedings of the 8th International Workshop on Structural Health Monitoring*, volume 2, pages 2173–2180, Stanford, CA, 2011.

[6] Do Van Giang, T. Taleb, K. Hashimoto, N. Kato, and Y. Nemoto. A fair and lifetime-maximum routing algorithm for wireless sensor networks. In *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, pages 581–585, 2007.

[7] Pai-Hsiang Hsiao, A. Hwang, H. T. Kung, and D. Vlah. Load-balancing routing for wireless access networks. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 986–995 vol.2, 2001.

[8] Raj Jain, Dah-Ming Chiu, and William R Hawe. *A quantitative measure of fairness and discrimination for resource allocation in shared computer system.* 1984.

[9] Tian Lan, D. Kao, Mung Chiang, and A. Sabharwal. An axiomatic theory of fairness in network resource allocation. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9, 2010.

[10] Philip Levis. Tinyos programming. *Available Online at `http://www.tinyos.net/tinyos-2.x/doc/pdf/tinyos-programming.pdf`*, 2006.

[11] P.H. Pathak and R. Dutta. Impact of power control on relay load balancing in wireless sensor networks. In *Wireless Communications and Networking Conference (WCNC), 2010 IEEE*, pages 1–6, 2010.

[12] Lucian Popa, Afshin Rostamizadeh, Richard Karp, Christos Papadimitriou, and Ion Stoica. Balancing traffic load in wireless networks with curveball routing. In *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '07, pages 170–179, New York, NY, USA, 2007. ACM.

[13] V. Rodoplu and T.H. Meng. Bits-per-joule capacity of energy-limited wireless networks. *Wireless Communications, IEEE Transactions on*, 6(3):857–865, 2007.

[14] Yangming Zhao, Sheng Wang, Sizhong Xu, Xiong Wang, Xiujiao Gao, and Chunming Qiao. Load balance vs energy efficiency in traffic engineering: A game theoretical perspective. In *INFOCOM, 2013 Proceedings IEEE*, pages 530–534, 2013.