This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2009 proceedings.

Lifetime Benefits through Load Balancing in Homogeneous Sensor Networks

Daniele Puccinelli
University of Applied Sciences of Southern Switzerland
CH-6928 Manno
Email: daniele.puccinelli@supsi.ch

Martin Haenggi
University of Notre Dame
Notre Dame, IN, USA 46556
Email: mhaenggi@nd.edu

***Abstract*—In routing protocols for wireless sensor networks energy efficiency is of paramount importance. Reliability-oriented protocols discard lossy links to avoid the significant energy cost of packet loss. The downside is that nodes with a particularly favorable channel tend to be overused: their lifespan is curtailed and the total amount of data delivered by the network may be significantly reduced. This problem is particularly critical for the nodes that provide access to the sink, since they have to carry the weight of the whole network. The use of load balancing schemes can be expected to provide significant lifetime benefits: rather than always using the nodes with the best channel, traffic is redistributed over a larger number of relays. We quantify the benefits of load balancing by comparing a routing protocol with embedded load balancing to a reliability-oriented protocol. We present and interpret experimental evidence of the benefits that stem from load balancing, but at the same time we also show that there are situations in which load balancing does not help.**

I. INTRODUCTION

A. Motivation

Energy-efficient data collection is vital in resource-constrained, lower-end wireless sensor networks (WSNs). At the network layer, reliability-oriented cost-based routing is a typical design choice [1] supported by the need to minimize packet loss and the related energy waste. On the other hand, consistently choosing the most reliable relays exacerbates the *hot spot* problem [2], whereby the nodes that function as gateways to the sink (often called *critical nodes*) experience a premature depletion of their energy reserves as they are called to forward many-to-one traffic from the upstream nodes. If the best relays are privileged by the routing protocol, then the lifespan of the critical nodes that offer the most reliable links to the sink is curtailed to an even greater extent. Load balancing schemes address this phenomenon. They can be applied at several levels, but our focus is on network-layer load balancing, *i.e.*, load balancing embedded in a routing protocol. The idea is to occasionally leverage on suboptimal links in alternative to the most reliable ones to ensure a more uniform usage of the energy resources across the network [3]. Always choosing the most reliable relays is a greedy form of data collection aimed at mining the network as much as possible; while using only reliable channels avoids energy waste due to packet loss, it also drains the resources of the most reliable relays.

B. Contributions

The focus of this paper is on low-power WSNs. We provide a detailed experimental analysis of the benefits of embedding load balancing into a routing protocol. We use Arbutus, introduced in [4], as our case study, because it provides the possibility of enabling network-layer load balancing by incorporating a load term in the routing cost metric. We compare Arbutus to a reliability-based routing protocol, MultiHopLQI[1], which we choose as a benchmark because it has been strenuously tested in mote-based networks and has already been used in several studies [5][6] and real-world deployments [7]. The comparison is performed on the basis of experimental results with lower-end motes on Harvard's MoteLab testbed [8]. Our main goal is to complement the study in [4], where the load balancing performance of the two protocols is *estimated* under static energy conditions: in the experimental work presented herein, battery depletion is emulated so that it becomes possible to actually *measure* the load balancing performance of the two protocols. The results presented in this paper confirm the validity of the experimental analysis in [4].

We use a generic application and assume a many-to-one data collection scenario where every node sends data packets to the sink at the same rate. This scenario choice, along with the heavily clustered nature of the MoteLab testbed (with built-in topological bottlenecks at the time when the experiments were run), are ideal for an adversarial protocol evaluation, *i.e.*, an evaluation of the protocols under very critical conditions that are close to a worst-case scenario. We assume the network to be fully homogeneous: all nodes have the same hardware and software resources. Likewise, all nodes share the same energy constraints, with the exception of the sink, which is assumed to have renewable energy resources.

We employ sans-serif capital letters for the individual sensing nodes and indicate a link between nodes A and B with the notation (A,B). Given a node A, node B is said to be a neighbor of A if $\pi_{\mathsf{A},\mathsf{B}} > 0$, which means that at least one packet is successfully received over (A,B) during the observation window. The sink is denoted as S. The current parent of A is denoted as $p(\mathsf{A})$.

[1] http://www.tinyos.net/tinyos-1.x/tos/lib/MultiHopLQI.

978-1-4244-2948-6/09/$25.00 ©2009 IEEE

II. LOAD BALANCING IN SENSOR NETWORKS

Load balancing schemes have been proposed at several levels, such as topology control, redundancy suppression, controlled mobility, and as part of routing protocols; hybrid solutions across these categories also exist. Redundancy suppression may be used to enhance virtually any load balancing solution (for instance in the form of data aggregation), but it requires very specific assumptions on the application. A hybrid solution between topology-control and routing is clustering-based routing. Within this category, Siphon [9] recognizes the challenge posed by the hot spot problem and proposes a practical solution (backed by experimental evidence) based on the use of virtual sinks, special nodes equipped with a secondary radio that serve as local safety valves for overload traffic management. Our focus, however, is on strictly homogeneous WSNs, so clustering and virtual sinks are not viable options. We also assume nodes to be completely static, which means that we cannot use controlled mobility either. We therefore focus on load balancing as part of routing protocols.

Routing in WSNs is typically cost-based [10], and the central idea is the generation of a cost field rooted at the sink, set up as nodes estimate the cost of reaching the sink according to a given metric, *e.g.*, the distance in number of hops, while route reply is achieved by means of *reverse path routing* (data packets descend the cost field from the sensing area to the sink). The cost-based paradigm is central to the MintRoute architecture [1], which can be classified as reliability-oriented cost-based routing (costs are assigned based on channel reliability). The MintRoute paradigm has led to several other protocols; MultiHopLQI is one of the most widely used.

Energy-Aware Routing (EAR) [11] adds lifetime-awareness to the cost-based paradigm. Residual node energy is taken into account for cost estimation, and route selection is performed by randomly choosing paths through the cost field with probabilities computed on the basis of residual energy levels (*i.e.*, paths with little residual energy are chosen with low probability).

Arbutus [4] draws inspiration from the MintRoute architecture; it takes active steps to keep the routes short and allows the active pursuit of load balancing by taking the relayed load into account for the selection of the next-hop neighbor. Like EAR, Arbutus is a lifetime-aware cost-based solution, but its routing choices are completely deterministic. Similarly to Arbutus, Traffic-Adaptive Routing (TAR) [12] takes traffic load into account for route selection. The key difference is that TAR uses geographic information, while Arbutus is cost-based.

Some similarities (mostly in terms of methodology) exist between this paper and [13], which deals with congestion control in WSNs and presents a cross-layer (transport-network-MAC) approach called Fusion. The authors of [13] manage their own testbed using the MoteLab interface and use it to evaluate their scheme under traffic conditions similar to the ones we consider.

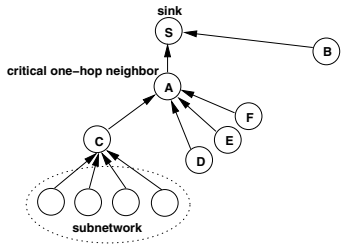

Fig. 1. Reliability-oriented routing tends to overexploit the nodes with the best channel to the sink, like A in this example.

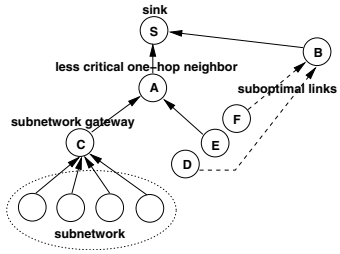

Fig. 2. Load balancing extends the lifetime of A with respect to the reliability-oriented approach by using suboptimal links.

III. EXPERIMENTAL SETUP

A. MultiHopLQI

The routing protocols commonly used by the TinyOS community are sender-based, address-centric, and employ a collection tree. MultiHopLQI employs the cost-based paradigm defined in [1], which recognizes that the volatility of the wireless channel makes the Boolean connectivity model unsuitable for use in lower-end sensor networks with low-power radios and limited resources. Link estimation is viewed as an essential tool for the computation of reliability-oriented route selection metrics. Routing is broken down into three major components: a link estimator that continuously assesses link quality, a routing engine that determines the address of the one-hop neighbor that provides the best progress toward the sink according to a given cost function based on the output of the link estimator, and a forwarding engine that injects its own traffic or relays upstream traffic by unicasting to the address determined by the routing engine. The rationale behind this partitioning is the separation of the data plane (forwarding engine) from the control plane (routing engine with the link estimator of choice). Connectivity discovery and route maintenance are carried out with the help of control beacons that diffuse global state used locally for route selection.

In MultiHopLQI the link metric is a particular form of Channel State Information, the Link Quality Indicator (LQI), used additively to obtain the cost of a given route. LQI is specific to IEEE 802.15.4-compliant devices[2]. MultiHopLQI avoids routing tables by only keeping state for the best parent at a given time, drastically reducing memory usage and control

[2]The IEEE 802.15.4 standard does not specify the implementation of LQI, which is up to the radio manufacturer.

overhead. A new parent is adopted if it advertises a lower cost than the current parent.

B. Arbutus

Arbutus [4] is closely related to the MintRoute family: it is also tree-based, cost-based, sender-based, and address-centric. As distinct to MintRoute and MultiHopLQI, Arbutus may be configured to compute routes based not only on reliability, but also on load, which is taken into account as part of the routing metric. The key difference between Arbutus and the MintRoute family lies in the tree construction process, based on the principle that routes with a low hop count are desirable [14], especially in networks as resource-constrained as lower-end WSNs. The main benefits of a low hop count are an increase in energy efficiency and end-to-end path reliability, a better load balancing, and a reduction in the route maintenance overhead.

Figure 1 shows a typical example of reliability-oriented routing in a WSN. Node A has a very large load due to the fact that it offers a particularly reliable link to the sink. At the same time, its upstream neighbor C is overloaded because it serves as a gateway node to a whole subnetwork (*i.e.*, it is the only way for the subnetwork to get to A and on to the sink). Figure 2 shows the approach taken by Arbutus: becoming aware of the fact that A is a load bottleneck, nodes D and F award node B with parent status even though link estimates indicate that their links to B are lossy. In the short run, the loss over the suboptimal links, in this case (D, B) and (F, B), may be offset by the congestion reduction at node A. More significantly, in the long run, the lifetime of node A is extended thanks to the load redistribution, which brings a long-term benefit in terms of delivered data (but curtails the lifetime of B). A coverage benefit is a less obvious yet frequent consequence of load redistribution: in this example, due to the reduced fan-in of A, it becomes easier for node C to forward its own packets as well as the ones generated by the nodes in the subnetwork, and there is a higher chance for the nodes in the subnetwork to be heard. Node A still overhears packets from D and B, but no longer wastes time and energy to process them.

C. Energy depletion emulation

Arbutus and MultiHopLQI have already been compared in [4], where it is shown that the relaying cost-to-benefit ratio of MultiHopLQI is reduced by 30%, *i.e.*, Arbutus can achieve the same data delivery performance with a reduced network load. This can translate into a significant lifetime gain: a reduced load implies energy savings for the network as a whole, but in particular for the critical nodes, whose workload is reduced insofar as allowed by the topology of the network. At the same time, if the per-node load is decreased, node failures are on average not as critical and the network benefits from an improved fault tolerance.

The main goal of our experimental evaluation is to verify that the reduction in the cost-to-benefit ratio indeed corresponds to a similar lifetime gain. Since all nodes in the MoteLab testbed are wall-powered, we need to emulate the energy depletion process (on all nodes other than the sink). The energy expenditure of the MCU is similar across protocols of comparable complexity, and sensing energy is obviously independent of the routing protocol. Therefore, we emulate lifetime evolution by only factoring in the radio-related energy expenditure. Every node is assigned the same energy credit X; a credit unit is lost for every packet sent or received. It is essential to note that *every packet counts*: both data packets and control packets are factored in, no matter whether they have been sent, received, or overheard. A packet sent by node A is *received* by node B if it can be correctly demodulated (as guaranteed, for instance, by a cyclic redundancy check) by B and B is indeed its intended receiver ($\mathsf{B} = p(\mathsf{A})$). A packet sent by node A is *overheard* by B if it can be correctly demodulated by B but B is not its intended receiver ($\mathsf{B} \neq p(\mathsf{A})$). Once a node has exhausted 95% of its credit, it warns its upstream neighbors (child nodes) of its upcoming energy depletion by advertising parent loss; this is done to avoid unnecessary packet losses on the part of the child nodes after an emulated depletion event at the parent.

D. System operating point

Since load balancing is most needed when the traffic load is high, we operate at a high traffic generation rate: each node in the network is a source and generates data packets at a rate f_{gen} of 4 packets per second. This causes a large amount of in-network interference as well as congestion; Arbutus and MultiHopLQI are purposefully set to operate in best-effort mode, with no retransmissions and no congestion control.

Our operating point causes congestion and in-network interference: not only are there lossy links, but also ingress drops, and therefore heavy packet loss is to be expected. We focus on the ratio between a few performance indicators (the figures of merit in III-E) that we measure as we run the two protocols back-to-back (to ensure reasonable continuity in the connectivity properties of the network). Given the constraints on MoteLab use (it is a shared resource with a user quota allocation policy), we employ a credit of $X = 2000$ packets for depletion emulation, which allows us to emulate network depletion in slightly less than 10 minutes.

E. Figures of merit

Lifetime: Following [15], we define network $\eta-$lifetime as the time between the inception of network operation and the moment when the sink ceases to receive sensor data from at least a fraction η of the nodes. Other noteworthy lifetime indicators are the minimum node lifetime (time when the earliest node depletion occurs) and the mean node lifetime. Since at the chosen operating point all nodes continuously send out packets, we normalize the lifespan of each node with respect to the lifespan of a node that only transmits its own packets, without ever relaying any packets on behalf of its peers. Due to the dynamics of Arbutus's cost field propagation, some nodes may begin to respond after a certain delay; for this reason, Arbutus's normalized lifespan values may be observed

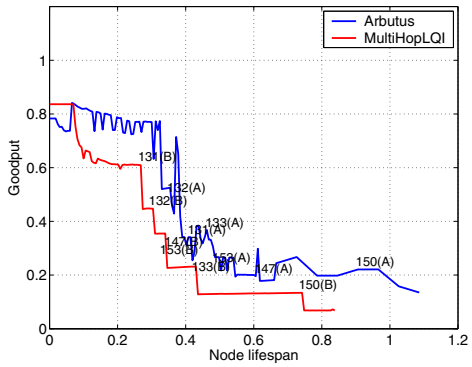

Fig. 3. Normalized overall goodput of Arbutus and MultiHopLQI on a 14-mote network (subset of the MoteLab testbed).

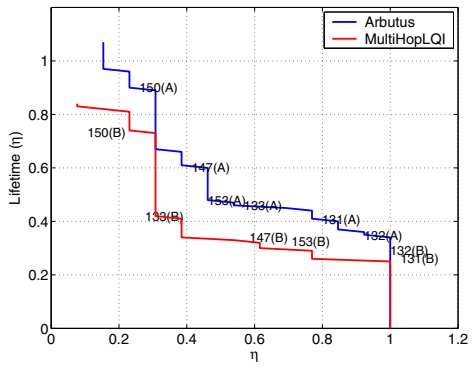

Fig. 4. The $\eta-$lifetime profile shows that individual nodes live substantially longer with Arbutus.

to exceed 1 (we ignore the maximum node lifespan values in our comparison, and focus on the minimum and the mean).

Total delivered data: The TDD is the total number of packets received by the sink, and is proportional to the network lifetime. The normalized overall goodput can be obtained as the ratio of the number of packets delivered to the sink over all the injected packets.

Sink neighborhood size: It is defined as the number of nodes with an average hop count (over a whole experiment) lower than 1.25 (we allow some leeway to account for hop count fluctuations).

Coverage: It is defined in this paper as the number of nodes from which the sink receives at least 100 packets (over the course of the emulated lifetime).

Control overhead: It is the ratio of the control traffic (received, sent, or overheard) over all traffic. Overheard data traffic is not control overhead and can be seen as a form of overhead induced by the network topology, since it is a byproduct of the broadcast effect. It is factored into the energy tally (see III-C), but not into the control overhead.

	Arbutus	MultiHopLQI
TDD [packets]	8691	6933
Coverage [nodes]	13	12
Minimum node lifetime	0.35	0.26
Mean node lifetime	0.69	0.39
Control overhead	0.05	0.10

TABLE I
OVERALL PERFORMANCE METRICS FOR THE SAMPLE RUN IN IV-A.

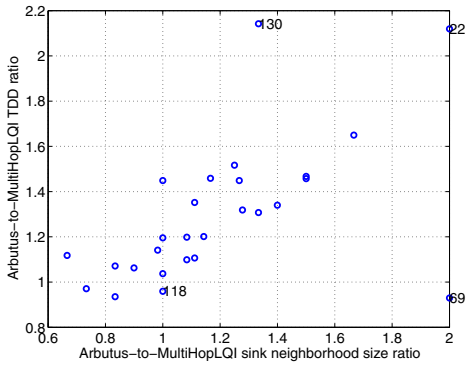

Fig. 5. The extension of the sink neighborhood is the main mechanism whereby Arbutus achieves a load balancing gain.

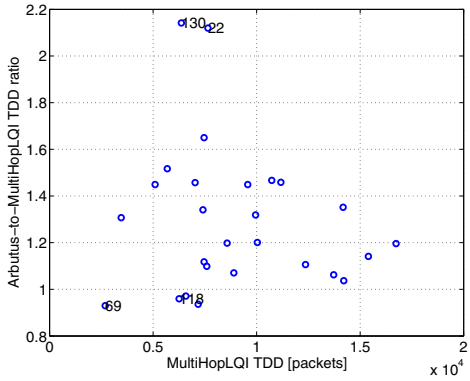

Fig. 6. MultiHopLQI only outperforms Arbutus in terms of TDD if scarce connectivity in the network causes heavy packet loss, thus reducing the traffic load.

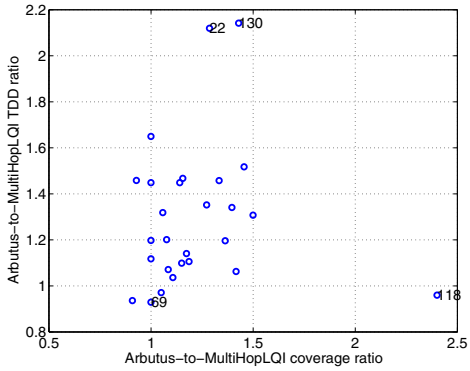

Fig. 7. Load balancing typically comes with a significant coverage benefit, as shown by this plot of the Arbutus-to-MultiHopLQI TDD ratio versus the coverage ratio.

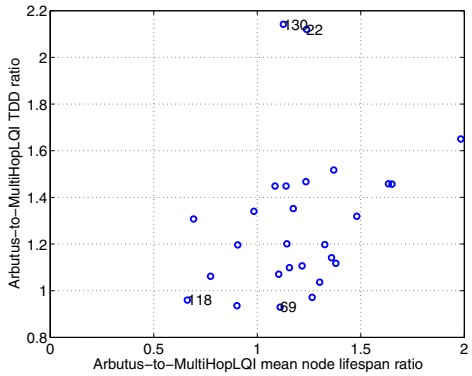

Fig. 8. It is intuitively pleasing that, as a general trend, an increase in TDD comes with an increase in the mean node lifespan.

A-M TDD ratio - average	1.30
A-M TDD ratio - standard deviation	0.31
A-M Coverage ratio - average	1.22
A-M Coverage ratio - standard deviation	0.29
A-M Earliest node depletion time ratio - average	1.18
A-M Earliest node depletion time ratio - standard deviation	0.50
A-M Mean node depletion time ratio - average	1.49
A-M Mean node depletion time ratio - standard deviation	0.41
A-M Control overhead ratio - average	0.87
A-M Control overhead ratio - standard deviation	0.23

TABLE II
OVERALL PERFORMANCE METRICS FOR OUR EXPERIMENTAL EVALUATION ON MOTELAB (A INDICATES ARBUTUS AND M INDICATES MULTIHOPLQI).

Fraction of experiments with TDD gain	0.81
Fraction of experiments with coverage gain	0.93
Fraction of experiments with coverage gain and TDD gain	0.78
Fraction of experiments with coverage loss and TDD gain	0.04
Fraction of experiments with TDD loss and coverage gain	0.15
Fraction of experiments with coverage loss and TDD loss	0.04

TABLE III
ARBUTUS PROVIDES BOTH A COVERAGE GAIN AND A TDD GAIN IN 78% OF OUR EXPERIMENTS.

IV. EXPERIMENTAL RESULTS

A. Sample run

To illustrate what happens with each protocol, we show an example of the performance of Arbutus and MultiHopLQI on the MoteLab testbed. The particular sink assignment is node 134, which lies within a relatively dense cluster that at the time of the experiment had virtually no connectivity to the rest of the network (as confirmed by point-to-point connectivity measurements taken prior to and after the routing experiments). Figure 3 shows the evolution of the normalized overall goodput as routing is underway and energy depletion is emulated. Qualitatively, what happens is that Arbutus distributes the load over all the nodes, while MultiHopLQI overloads the nodes that offer the best channel to the sink, namely 131 and 132, which have the shortest lifetime along with nodes that depend on them for connectivity to the rest of the network. Figure 4 shows the value of the $\eta-$lifetime as the function of the fraction η of responding nodes. Given a value of η, we can read the normalized time that a fraction η of the 13 nodes in the network respond to the sink. By extending the lifespan of the individual nodes, Arbutus always has a better or equal $\eta-$lifetime profile with respect to MultiHopLQI. More details can be found in Table I. The 54% TDD gain of Arbutus over MultiHopLQI is considerable, and is reflected by a significant lifetime gain: the mean node lifetime is increased by 77%, and the earliest depletion time for a node (minimum node lifetime) is up by 35%. The 50% reduction of the control overhead is a considerable benefit of the long-hop routing approach of Arbutus.

B. Performance evaluation

We chose 25 different sink assignments and ran a total of over 50 experiments on MoteLab. We average values from different experiments with the same sink assignment, and finally take the sample mean of the average values for the individual sink assignments, which are shown throughout this section. We express our figures of merit as the ratio of Arbutus's performance over MultiHopLQI's. Table II shows the results of the comparison of Arbutus and MultiHopLQI. The main result is that, on average, the use of load balancing increases the TDD by 31%. Since TDD is proportional to network lifetime, this result confirms the estimate in [4]. The average coverage gain is also significant, at 22%.

The lifetime benefits are significant: on average, load balancing delays the earliest depletion event by 18% and increases the life expectancy of a node by as much as 49%. Only nodes that do get depleted are considered (nodes that are far enough from the sink may not get depleted at all because their gateways to the sink get depleted first, depriving them of a route to the sink). There is a relatively large variance in the results: the standard deviation is about 0.3 for both the TDD and the coverage ratio. Table III shows a breakdown of the performance results based on what particular benefit (TDD or coverage gain) is or is not achieved. With most sink assignments, Arbutus provides both a TDD and a coverage gain.

Given a sink assignment, performance variations still occur because of network topology variations due to node failures or changing multipath fading patterns. The performance variations over different sink assignments are, however, much more dramatic. In Figures 5-8, we represent various flavors of the principal figure of merit, the Arbutus-to-MultiHopLQI TDD ratio (averaged over all datasets with the same sink assignment). To facilitate our interpretation, the particular sink assignment is shown for four outliers that stand out in different ways. While 22 and 130 always correspond to an outstanding performance (with respect to TDD, coverage, and node lifespan), 69 and 118 are examples of undesired behaviors on the part of Arbutus, and will be the focus of our discussion. Figure 5 shows that there exist a correlation between sink neighborhood size and TDD: the more Arbutus extends the number of sink's neighbors, the larger its TDD gain with respect to MultiHopLQI. The sink's neighborhood contains the

aforementioned critical nodes that have to carry the weight of the whole network as they are called to relay data packets from their upstream peers. Increasing the number of such critical nodes for load redistribution (to make them less critical) means using suboptimal links to reduce the hop count of some routes, which may pay off through the decongestion of the best relays. The presence of an outlier, 69, shows a slight TDD loss despite the doubling of the size of the critical area; this is an example where the use of suboptimal links does backfire. Figure 6 shows the Arbutus-to-MultiHopLQI TDD ratio as a function of the TDD of MultiHopLQI. It is not surprising to discover, in Figure 6, that sink assignment 69 corresponds to the smallest TDD over all the sink assignments that we employ (in this case the TDD is even smaller with Arbutus, as shown by the Arbutus-MultiHopLQI TDD ratio below 1 in Figure 5). Due to the network topology seen from node 69 (with the particular connectivity conditions of the testbed when the experiments were run), sink assignment 69 only yields less than half of the average TDD over all sink assignments. Under such lighter traffic, the critical area is not so critical and the use of suboptimal links becomes detrimental. To further prove our point, all other sink assignments that cause Arbutus to incur in a slight TDD loss lie at low volumes of sink-bound traffic (*e.g*, node 118).

Figure 7 confirms Table III by showing that a coverage gain usually comes with a TDD gain. With outlier 118 we have a relatively small volume of acquired packets (see Figure 6) but no sink neighborhood extension (see Figure 5): although Arbutus's tree structure improves coverage, the topology is the bottleneck, and the additional packets from the extra nodes that get covered drain the very limited relaying resources of the critical area. Figure 8 shows the TDD ratio as a function of the mean node depletion time. It is not surprising to find that Arbutus has the worst lifetime performance with sink assignment 118. As a general trend, an increase in the mean node lifespan corresponds to a larger TDD, which is intuitively pleasing and comes as a confirmation of the long-term benefits of load balancing.

V. CONCLUSIONS

Our experimental results show that embedding load balancing at the network layer provides significant lifetime advantages with respect to standard reliability-oriented solutions. Such advantages often translate into an enhanced total data delivery rate and increased coverage. In the long run, load balancing allows to save resources for future use. Load balancing is more critical in static environments: in the presence of changing multipath fading patterns, a reliability-oriented protocol is forced to perform some amount of parent rotation, which is not the case if fading is completely static (*i.e.*, it is essentially the product of the layout of the deployment area and it is not induced by human activity).

Particular topological conditions may cause critical bottlenecks that may not be avoided by way of load balancing. Similarly, particular topology conditions may also result in the cutoff of a large section of the network from the point of view of a given sink assignment, thus reducing the traffic load in the sink neighborhood. In these cases, load balancing does not help, and a reliability-based approach is preferable.

ACKNOWLEDGMENTS

The authors would like to thank Geoff Werner-Allen and Matt Welsh for making their MoteLab testbed available to the community. The support of NSF (CNS 04-47869) and CRANE/DTRA (N00164-07-8510) is gratefully acknowledged. A part of this work has been developed within the EU-NEST project MEMORY (FP6 EU-NEST 43236), whose support is also gratefully acknowledged.

REFERENCES

[1] A. Woo, T. Tong, and D. Culler. Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks. In *1st International Conference on Embedded Networked Sensor Systems (SenSys'03)*, Los Angeles, CA, USA, November 2003.

[2] M. Haenggi. Energy-Balancing Strategies for Wireless Sensor Networks. In *IEEE International Symposium on Circuits and Systems (ISCAS'03)*, Bangkok, Thailand, May 2003.

[3] J. Chang and L. Tassiulas. Energy Conserving Routing in Wireless Ad Hoc Networks. In *Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'00)*, Tel Aviv, Israel, March 2000.

[4] D. Puccinelli and M. Haenggi. Arbutus: Network-layer load balancing for wireless sensor networks. In *IEEE Wireless Communications and Networking Conference (WCNC'08)*, Las Vegas, NV, USA, March 2008.

[5] M. Wachs, J. Choi, J. Lee, K. Srinivasan, Z. Chen, M. Jain, and P. Levis. Visibility: A New Metric for Protocol Design. In *5th ACM Conference on Embedded Networked Sensor Systems (SenSys'07)*, Sydney, Australia, November 2007.

[6] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis. Four-Bit Wireless Link Estimation. In *Sixth Workshop on Hot Topics in Networks (HotNets-VI)*, Atlanta, GA, USA, November 2007.

[7] G. Werner-Allen, K. Lorincz, M. Welsh, O. Marcillo, J. Johnson, M. Ruiz, and J. Lees. A Binary Feedback Scheme for Congestion Avoidance in Computer Networks. *IEEE Internet Computing*, 10(2):18–25, March 2006.

[8] G. Werner-Allen, P. Swieskowski, and M. Welsh. MoteLab: a Wireless Sensor Network Testbed. In *4th international symposium on Information Processing in Sensor Networks (IPSN'05)*, Los Angeles, CA, USA, April 2005.

[9] C. Wan, S. Eisenman, A. Campbell, and J. Crowcroft. Overload Traffic Management in Sensor Networks. *ACM Transactions on Sensor Networks*, 3(4), October 2007.

[10] R. Poor. Gradient Routing in Ad Hoc Networks. 2000. Available at http://www.media.mit.edu/pia/Research/ESP/texts/poorieeepaper.pdf.

[11] R. Shah and J. Rabaey. Energy aware routing for low energy ad hoc sensor networks. In *IEEE Wireless Communications and Networking Conference (WCNC'02)*, Orlando, FL, USA, March 2002.

[12] M. Fyffe, M. Sun, and X. Ma. Traffic-Adapted Load Balancing in Sensor Networks Employing Geographic Routing. In *IEEE Wireless Communications and Networking Conference (WCNC'07)*, Hong Kong, China, March 2007.

[13] B. Hull, K. Jamieson, and H. Balakrishnan. Mitigating Congestion in Wireless Sensor Networks. In *2nd ACM Conference on Embedded Networked Sensor Systems (SenSys'04)*, Baltimore, MD, USA, November 2004.

[14] M. Haenggi and D. Puccinelli. Routing in Ad Hoc Networks: A Case for Long Hops. *IEEE Communications Magazine*, 43, October 2005.

[15] X. Liu and M. Haenggi. Towards Quasi-Regular Sensor Networks: Topology Control Algorithms for Improved Energy Efficiency. *IEEE Transactions on Parallel and Distributed Systems - Special Issue on Localized Communication and Topology Protocols for Ad Hoc Networks*, 17:975–986, September 2006.