Alternating Multiple Tributaries+Deltas

Goce Trajcevski[*]
Department of EECS
Northwestern University

Oliviu Ghica
Department of EECS
Northwestern University

Peter Scheuermann[†]
Department of EECS
Northwestern University

goce,ocg474,peters@eecs.northwestern.edu

Roberto Tamassia[‡]
Department of CS
Brown University
rt@cs.brown.edu

Isabel F. Cruz[§]
Department of CS
University of Illinois at Chicago
ifc@cs.uic.edu

ABSTRACT

This work addresses the problem of trading off the latency of delivering the answer to the sink, at the benefit of balancing the spatial dispersion of the energy consumption among the nodes and, consequently, prolonging the lifetime in sensor networks. Typically, when a query is posed pertaining to the data from some geographic region, typically a tree-structure is constructed and, when possible, some in-network aggregation is performed. On the other hand, in order to increase the robustness and/or balance the load, multipath routing is employed. Motivated by an earlier work that combined trees and multipaths [18], in this paper we explore the possibility, and the impact, of combining multiple trees and multiple multipaths for routing, when processing a query with respect to a given region of interest. We present and evaluate two approaches that enable load-balancing in terms of alternating among a collection of routing structures.

1. INTRODUCTION

Various aspects of the problem of routing in wireless sensor networks settings have received considerable attention [2], due to the fact that the communication has the largest impact on the depletion of the batteries of the sensor nodes. Multipath routing approaches [10, 9, 15, 20, 23, 22] have been mostly used for two purposes: load-balancing (which, in turn, prolongs the lifetime of the network) and robustness of the transmission. Tree-based routing structures, on the other hand, have also been studied extensively and from different perspectives, e.g., in-network processing of aggregate queries [16, 17]; maximizing geographic knowledge [24]; balancing workloads [3, 18]; and exploiting knowledge about the mobility of sinks [26, 27].

An approach that combines tree-based and multipath-based routings, called "Tributaries and Deltas", was presented in [18]. The focus of this work was to adjust the balance between trees ("Tributaries") and multipaths ("Deltas") in response to varying network conditions, expressed in terms of the packets drop rate, for the purpose of robust and efficient in-network computation of aggregates. In particular, algorithms are presented in [18] for computing frequent itemsets and quantiles along with the criteria for changing the *role* of a particular node (e.g., from being a node into a tree, into becoming a node in a multipath).

Motivated by this, we set to explore the benefits that a given sensor network may gain when *multiple trees* are used in conjunction with *alternating (collections of) multipaths* for routing the results of a given query $\mathcal{Q}$ pertaining to a geographic region Q_R. Specifically, we present two complementary approaches and investigate their impact on the load balancing in terms of the spatial distribution of the energy-consumption of the nodes involved in processing $\mathcal{Q}$. An example illustrating our problem-settings is given in Figure 1. Figure 1.a presents a scenario in which the readings of the sensors in a region Q_R are transmitted to a given sink. Here, the routing is executed using a tree combined with multiple k-short based (cf. [14, 9, 23]) paths. Figure 1.b illustrates another tree, accompanied by another family of multiple paths that can be used for processing the same query. A particular tree determines the family of routes to be used outside Q_R and towards the sink, however, combining them in an alternating-in-time manner can increase the lifetime of each of the nodes involved. Finally, Figure 1.c presents the other alternating policy that we investigate – splitting the set of nodes in Q_R into two subsets that will be used to form two trees that will operate concurrently.

The rest of this paper is structured as follows. In Section 2 we present the necessary background. Section 3 gives a detailed description our approach. Section 4 reports our ex-

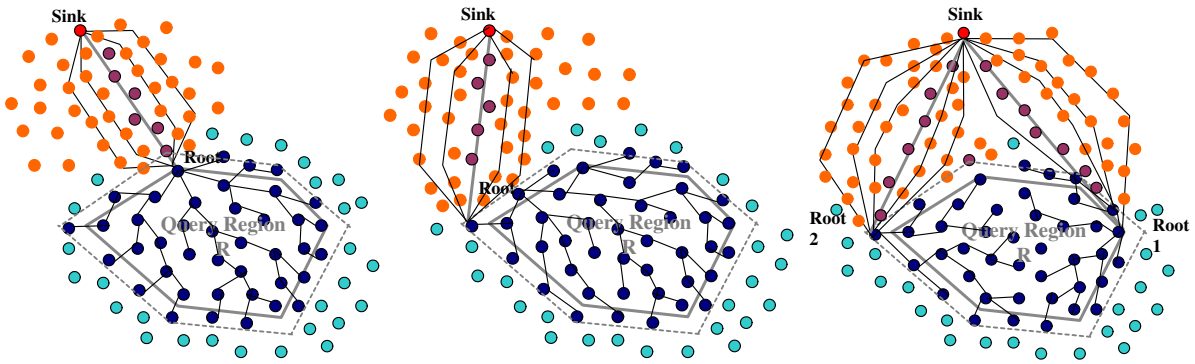
[*]Research supported by the Northrop Grumman Corp., contract: P.O.8200082518

[†]Research supported by the NSF grant IIS-0325144/003

[‡]Research supported in part by NSF grants IIS–0324846, IIS–0713403 and OCI–0724806

[§]This work was partially supported by NSF Awards ITR IIS-0326284 and IIS-0513553.

DMSN'08 August 24, 2008, Auckland, New Zealand



a.) Shortest multipath b.) Alternative multipath + tree c.) Multiple trees and multipaths

Figure 1: Multiple trees and multipaths

perimental observations. In Section 5 we position our work with respect to the relevant literature, and in Section 6 we summarize and outline directions for future work.

2. PRELIMINARIES AND SYSTEM MODEL

We assume that the sensor network consists of N nodes, $SN = \{S_1, S_2, \ldots, S_N\}$ and each node is aware of its location $S_k = (x_{Sk}, y_{Sk})$, either through a GPS or by using some other technique (e.g., trilateration based on a few beacon-nodes) [5]. In addition, each node knows the location of its one-hop neighbors, i.e., the nodes within its communication range. The nodes are assumed to be static, that is, their geographic locations do not change over time.

A particular query, $\mathcal{Q}$ is specified as a quintuple $(Sink, Q_R, S_{val}, F, \Delta, Type)$, where:
(1) $Sink$ is the ID (and the known location) of the sink-node, which is the final destination of all the packets containing values/readings of relevance for $\mathcal{Q}$;
(2) Q_R is the geographic region of interest, from which the readings of a particular value(s) is needed. We assume that it is bounded by a polygon and is represented as a sequence $\{(x_{v1}, y_{v1}), (x_{v2}, y_{v2}), \ldots, (x_{vn}, y_{vn})\}$ of its vertices in a counter-clockwise order;
(3) S_{val} describes the particular value that needs to be monitored (e.g., *Temperature*);
(4) F denotes the sampling frequency for the query;
(5) Δ specifies the temporal duration, in terms of an interval $[t_{begin}, t_{end}]$;
(6) The parameter *Type* is a string that specifies the query operation performed like, for example, monitoring some aggregate value (e.g., SUM, COUNT, AVG) [16, 17, 18] or simply gathering the readings of individual nodes.

When it comes to constructing a routing tree rooted at a particular node, we assume that some of the standard geographic routing protocols [28], e.g., similar to the ones used in TAG approach [16, 17], are used. Likewise, we rely on the existing works for constructing a multipath routing structure (e.g., SMR [14]) for a given pair of nodes. Depending on the application requirements, these paths may be completely (edge and node) disjoint or not (e.g., braided [9]). Often, a collection of multiple paths between a given *(source, sink)* pair is constructed based on the *k-short* methodology [14], and we explain the minor modifications that we adopt for the purpose of load balancing in Section 3.2. Lastly, we assume that sink node S_{sink} is outside Q_R, and distant enough so that it is a vertex of the convex hull of $\{(x_{v1}, y_{v1}), (x_{v2}, y_{v2}), \ldots, (x_{vn}, y_{vn})\} \cup \{S_{sink}\}$.

3. CONSTRUCTION OF ALTERNATING TRIBUTARIES AND DELTAS

In this section, we present the key ideas of our proposed methodology. First, we explain the basic steps needed for generation and propagation of a particular request from the sink towards the query-region Q_R. Subsequently, we discuss the details of each of the alternating approaches:
(1) The first one generates a *family of trees* rooted at nodes within a given distance δ from the boundary of Q_R. These trees are used in a (permuted) sequence and each of them, in turn, uses a different collection of k-short paths, thus balancing the workload among the nodes that participate in multipaths outside Q_R;
(2) The second one partitions the nodes inside Q_R in a manner that yields two trees to route the sensed data in parallel. Lastly, we discuss some practical issues regarding the management of the multipaths.

3.1 Initialization

The initial setup for processing a particular query—generation of the request along with its propagation—consists of the following three phases:

(Phase I:) The sink node S_{sink}) transmits an RREQ packet for initiating the processing of the query, which contains the following information:

1. Its own location $S_{sink} = (x_S, y_S)$.
2. The coordinates of the vertices of the polygon bounding Q_R.
3. The location of the point $B_C = (b_{cx}, b_{cy})$ on ∂Q_R, the boundary of Q_R, which is *closest* to the sink. This information implicitly determines the (shortest path) route to be used for the TBF-based forwarding of the RREQ [19].
4. The deviation-tolerance parameter δ, which specifies *how far* from the edges of the polygon bounding the region of interest Q_R is the locations an individual

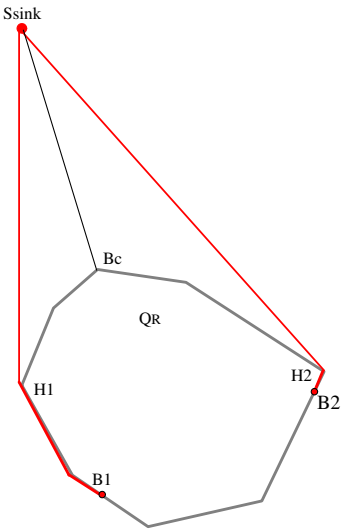


Figure 2: Boundaries for the roots of alternating trees

sensor, say S_l, allowed to be, in order for it to participate in the query. This is needed because the locations of the sensors need not be exactly along ∂Q_R, so some tolerance needs to be specified in the sense that $\|S_l, \partial Q_R\| \leq \delta$, where $\|\cdots\|$ denotes the Euclidean distance (cf. [6]).

5. The extra latency in the transmission that S_{sink} is willing to tolerate, in terms of the additional delay incurred when the packets containing the results of the query are *not* transmitted via the shortest path from (the root of the tree in) Q_R to the sink. This is expressed by two additional points B_1 and B_2 ($\in \partial Q_R$) whose locations are transmitted as part of the RREQ. To determine those locations, S_{sink} finds the points on ∂Q_R such that $geo_{dist}(S_{sink}, B_i) = \kappa \|S_{sink}, B_C\|$ ($i \in \{1, 2\}$), where geo_{dist} denotes the *geodesic* distance [11]. Hence, κ is the constant which specifies what is the acceptable tolerance for the increase of the latency, with respect to the one using the TBF-based shortest path. An illustration is provided in Figure 2, which shows that the geodesic distance between B_1 and S_{sink} is the sum of $\|S_{sink}, H_1\|$ and the length of ∂Q_R between H_1 and B_1, where H_1 is the adjacent node to S_{sink} in the convex hull of $\{(x_{v1}, y_{v1}), (x_{v2}, y_{v2}), \ldots, (x_{vn}, y_{vn})\} \cup \{S_{sink}\}$. Note that it may be the case that neither B_1 nor B_2 are on the outer boundary of the convex hull. The computational overhead at S_{sink} is dominated by the construction of the hull ($O(n \log n)$).

In addition, RREQ contains all the other parameters of relevance for the query (i.e., $F, \Delta, Type$).

(Phase II) Once the values of all the attributes have been determined, the RREQ packet from S_{sink} is forwarded along the trajectory defined by the line-segment $\overline{(x_S, y_S)(b_{cx}, b_{cy})}$, which is the shortest path from the (locations of the) sink to ∂Q_R. The actual forwarding methodology can be any of the well-studied geo-routing approaches, e.g., TBF-like [19], incorporating face-routing GOAFR+ [13] to alleviate the problem of greedy forwarding and/or energy awareness [23]. In addition to the local selection for the next hop along the route for forwarding the RREQ, each sensor node $S_i = (x_{si}, y_{si})$ now performs some extra processing:

1. If the geographical location of $B_C = (b_{cx}, b_{cy})$ is within its communication range, it selects the neighbor $S_j = (x_{sj}, y_{sj})$ for which the distance $\|B_C, S_j\|$ is smaller than the distance from B_C to any other node within S_i's range, as the next hop. Note that S_i itself may be such node.

2. Once S_j has been selected, the RREQ is transmitted to it, at which point S_j checks whether its distance from ∂Q_R is $\leq \delta$. If not, S_j repeats the first step of Phase II, until a node is found that is closer than itself to B_C and within the region $\partial Q_R \oplus \delta$, where $\oplus$ denotes the Minkowski sum operator [8].

(Phase III) After the selection of the S_j, the last phase, which consists of two basic tasks, focuses on properly incorporating the nodes inside Q_R:

1. The root (S_j) propagates the request for constructing a tree in the interior of Q_R and rooted at itself (c.f., [24]).

2. The vertices of Q_R and the tolerance parameter δ are also propagated. Parameter δ is used by the nodes that receive the request for the construction of the tree rooted at S_j, so that they can determine, based on their location, whether they should participate in processing the query. Since the nodes need not be located exactly on the boundary of Q_R [6], a node will decide whether it participates in the processing of query $\mathcal{Q}$ depending on whether it is within distance δ from ∂Q_R.

3.2 Multiple Alternating Trees

When determining the locations of the possible roots of the (collection of) alternative trees within $\partial Q_R \oplus \delta$, there is an important issue that needs to be considered. Namely, it has been observed that the nodes within the vicinity of the sinks/roots are the most heavily exploited in long-running queries [25]. Hence, it is desirable that there is some criteria that will bound the extent of the overlap between the nodes in the vicinities of the roots of any two "neighboring" trees. It is beyond the scope of this paper to investigate the optimization problem for the minimal energy consumption as a function of the overlap of the nodes. However, one criterion that we use is the degree of a *level-overlap* between two consecutive routing trees, which is defined as follows:

Given two trees $Tr_1(root_1)$ and $Tr_2(root_2)$, where $root_1 \neq root_2$, their level_i overlap is the set of nodes (sensors) that are at level i in both Tr_1 and Tr_2.

We say that two trees have a *level_i separation* if their *level_i overlap* is empty.

Finally, we need to determine the boundaries of the possible locations of the roots of the alternating trees, i.e., the *last-possible roots*. Recall that, as part of the query's specification, S_{sink} transmits the two locations B_1 and B_2 on ∂Q, which represent the limits of its tolerance for the delay. Hence, the last possible node that could be a root of any alternating tree is the node which is closest to B_1 (resp.

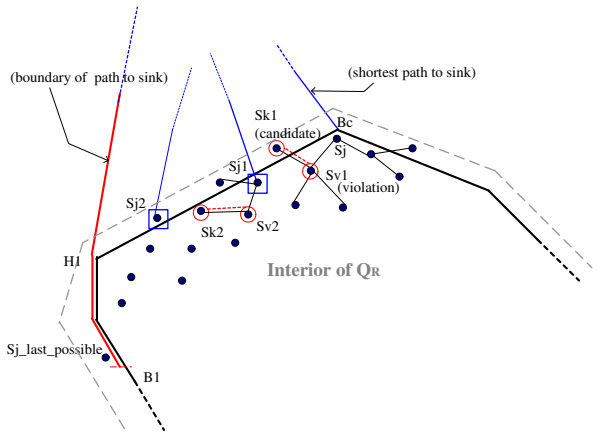


Figure 3: Selection of roots for alternate trees

B_2) *and* its closest point on ∂Q_R is on the portion of the perimeter between H_1 and B_1 (resp. H_2 and B_2).

A scenario illustrating the main concepts that we have introduced thus far is provided in Figure 3. The root of the initial tree (S_j) is the node that is closest to B_C, which is the node on ∂Q_R that is closest to the (location of the) sink in terms of the Euclidean distance. The leftmost part of the Figure 3 illustrates the geodesic distance used to determine the location of the node that is a last candidate for a root of some of the alternate trees, denoted by $S_{j\_last\_possible}$. As shown, this node is actually the one that is closest to the location of the point B_1, calculated in Phase I as a furthest-possible root, based on the tolerance κ (cf. Section 3.1). The node denoted as S_{k1} in Figure 3 is a candidate for a root of some alternative routing tree for Q_R, however, it violates the *level_1 separation* criteria, assumed for this example, because the node S_{v1} is at *level_1* in both trees (the original one, rooted at S_j, and the one that would have been rooted at S_{k1}). Hence, the next candidate for the root of an alternate tree in a counter-clockwise traversal direction of the boundary of Q_R is the node denoted by S_{j1}. The subsequent candidate, S_{k2} is eliminated in a similar manner, which makes S_{j2} the root of the next alternative tree.

3.3 Parallel Transmission with Disjoint Trees

An additional approach for load balancing that we consider is the construction of disjoint trees in Q_R. Initially, we focus on constructing two such trees and, for that purpose, we need to partition the query region into two subsets that contain (approximately) an equal number of nodes.

For brevity, assume that the nodes are uniformly distributed in the geographic region, which enables us to reason about the partitioning in terms of equal areas. An illustration of our method is provided in Figure 4:
(1) Divide Q_R into two regions, separated by the line-segment $\overline{B_1 B_2}$;
(2) color one of the regions *red* and the other *blue*;
(3) Draw a separator line which splits the red and the blue areas in half.

The above procedure (i.e., the separator line in step (3)) is an instance of the red-blue *ham-sandwich cut* problem [12] for which solutions also exist for sets of discrete "red" and "blue" points, in case one needs to balance the number of individual sensor nodes. The computational complexity of a

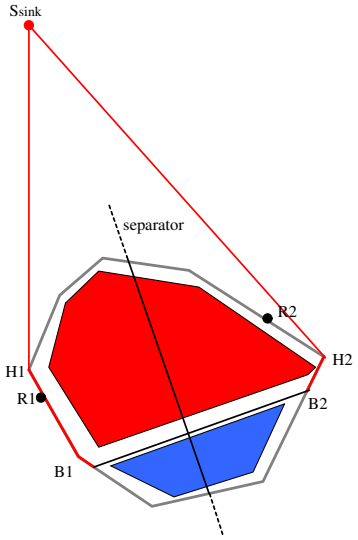


Figure 4: Disjoint Trees

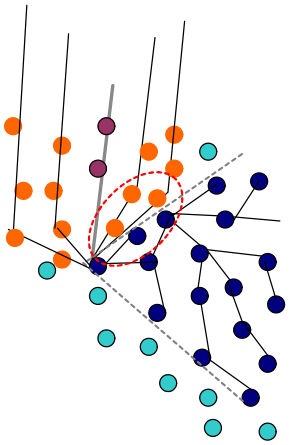


Figure 5: Multipath avoiding Tree-nodes

ham-sandwich cut for two convex polygons with a total of n nodes is ($O(n)$). For two arbitrary polygons, a *geodesic ham-sandwich* cut will be applied [4] and if two or more disjoint trees are desired, the results in [21] will be used. However, we leave the details of this extension for future work.

3.4 Some Practical Considerations

An important observation for load balancing purposes is that, in some cases, in addition to being disjoint within a particular collection of multipaths [9, 14], the construction of the routes will have to exclude the nodes that are part of a routing tree. An example is provided in Figure 5, which is a zoomed-out version of a portion of Figure 1 b.).

There are few other aspect that need to be addressed:
(1) How often should the alternating among different (*tree, multipath*) combinations be done? Clearly, in order for each such pair to be equally used, which is the goal of load-balancing, the minimum number of alternations is $\lfloor \frac{\Delta}{K} \rfloor$. On the other hand, the alternation among different trees may be done as often as once per each epoch [16]. A lower number of alternations is preferred because it provides the opportunity

to prolong the sleeping period of the nodes [7]. However, determining the sleeping schedule and coordinating it with the alternating frequency is a parameter that depends on the node loads, in terms of their involvement in processing other queries.
(2) How should the sequencing among different (*tree, multi-path*) be orchestrated? Intuitively, one needs to avoid consecutively selecting two trees whose roots are spatially-close for the purpose of minimizing the interference during the switching time.
(3) One last observation pertains to the value of parameter κ. Namely, we have used it mostly for determining the boundaries of the possible locations for the roots along ∂Q_R. However, in reality, a similar notion is needed as a "delay-bound" on the longest route in a given multipath. Hence, a more detailed analysis would involve splitting κ into two values κ_1 for the trees and κ_2 for the routes in a multipath collection ($\kappa = \kappa_1 + \kappa_2$).

4. EXPERIMENTS

For the purpose of experimental evaluation of our proposed methodology, we used SIDnet-SWANS,[1] an open-source wireless sensor network simulator. The testbed consisted of 500 nodes randomly distributed using a uniform distribution function in an area of $6,000 \times 6,000$ sqft. Nodes are homogeneous, sharing the same configuration: 40000 bps transmission/reception rate and power consumption characteristics that are based on the Mica2 Motes (c.f. [1]). We assumed that each node is powered by a $75mAh$ battery, for an expected average depletion interval, under load, of up to 72 hours. Motes are assumed to have capability of powering off their radio transceivers in order to preserve energy and we have averaged the results over three sleeping schedules consisting of $1,000ms$, $200ms$ and $20ms$ time-to-sleep intervals. For each of the experiments we randomly generated a bounding region for the query (5 rectangles and 5 pentagons) and an outside location for the corresponding sink. The only constraint that we imposed was that the number of alternating routing trees inside the region is between 5 and 11. The sampling frequencies that we used as parameters of our experiments (over which, again, we averaged) were: $0.5s$, $1s$ and $4s$.

All the experiments were conducted on an Intel Pentium Core 2 Duo Extreme machine with 2GB of DDR2 RAM.

Figure 6 illustrates the benefits obtained when a k-short based alternating collection of paths is used instead of a single path. Such benefits have already been studied in the literature and we report them here for the purpose of comparison with the extensions in the nodes' lifetime obtained via multiple trees combined with alternating paths.

Figure 7, accompanied with the tabular values, illustrates the benefits of the approaches proposed in this paper. It depicts the minimum residual energy depletion rate over the entire network, which, not surprisingly, corresponds to the energy depletion rate of the root nodes. As it can be observed, performing load-balancing by means of disjoint trees represents a better choice than alternating multiple trees, both of which outperform the "basic" case of having a single tree and one alternating path collection.

The last experiment that we conducted measures the lifetime of the network according to three different definitions (see Figure 8). Once again, both of the proposed methodologies yield improvements over the single tributary+delta approach and the disjoint trees appear to be the superior choice.

The experiments reported in Figures 7 and 8 confirm that the nodes in the vicinity of the root(s) of the aggregation tree(s) are the biggest energy consumers and represent the best candidates for energy-balancing techniques.

[1]SIDnet-SWANS is publicly available at *http://www.eecs.northwestern.edu/∼peters/sensors.*

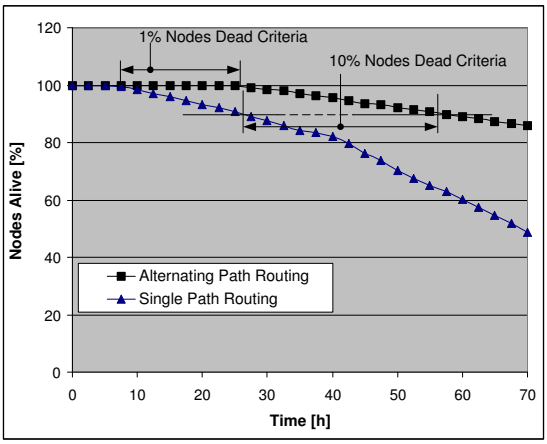
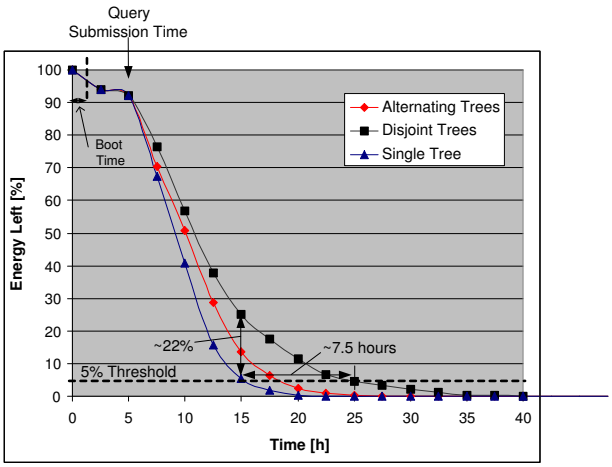


Figure 6: Lifetime: single vs. alternating multi-paths.



Time	0h	5h	10h	12.5h	**15h**	**17.5h**	20h	22.5h
Disjoint Trees	100	92	56.77	37.95	27.20	**17.46**	11.55	6.76
Alternating Trees	100	92	50.88	28.67	13.48	**6.47**	2.55	0
Single Tree	100	92	40.88	15.77	5.41	1.71	0	0

Figure 7: Minimum residual energy depletion rate over the network

5. RELATED WORK

There is a large body of work addressing both multipath routing [10, 9, 14, 15, 20, 23, 22] as well as tree routing [3,

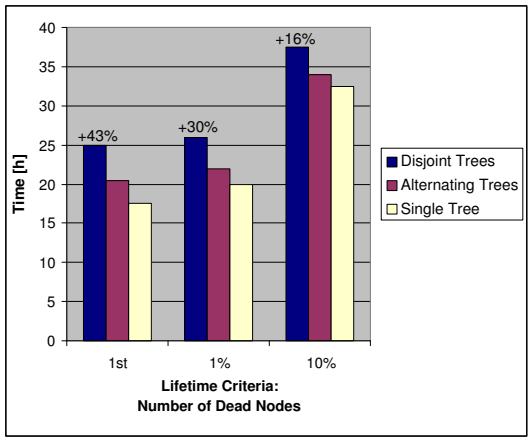


Figure 8: Lifetime of the network

26, 27, 16, 17, 24]. A comprehensive survey is presented in [2].

In its basic nature, the process of constructing the different routes in our work belongs to the category of geographic-based routing [28], relying on trajectory-based forwarding [19]. However, the observations regarding our proposed methodologies can be carried over to other (e.g., topological) approaches.

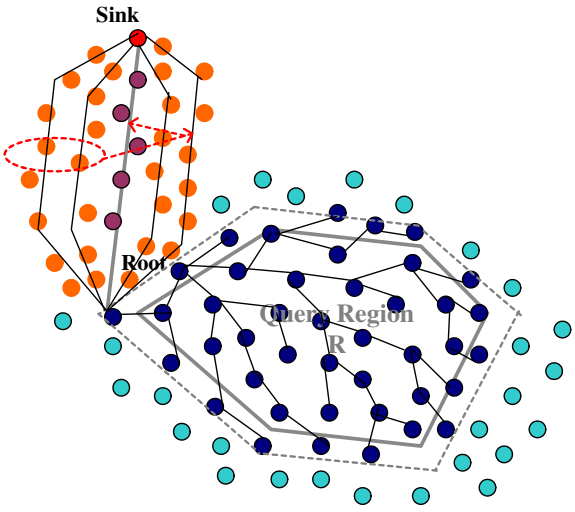


Figure 9: Alternating among multipaths

As we already indicated, the work that motivated the results presented in this paper, is the one of Tributaries and Deltas [18], where a combination of trees and multipaths was considered from the robustness perspective. However, that approach focused on adjusting the geographical regions in which the trees are used vs. the ones in which multiple paths are used, for the purpose of processing of aggregate queries in the presence of variable network conditions (e.g., link failure). Our work is, in a sense, complementary to [18] because it investigates the lifetime benefits obtained via using multiple trees and a collection of multiple routes (one for each tree) along with the packet-loss.

6. CONCLUSIONS AND FUTURE WORK

We have investigated the problem of load balancing for the purpose of prolonging the operational lifetime of the nodes in a sensor network, at the expense of an additional (bounded) delay, as a quality-of-data criteria, by means of combining multiple trees and multiple multipaths. We have proposed two methodologies for: (1) generating a collection of trees that can be used in alternation; and (2) constructing disjoint trees that can be used concurrently—each providing the same spatial coverage for a given query-region, but yielding better energy dissipation than a single tree.

An important problem that we plan to address in the future is how to coordinate the alternation with the robustness requirement, within a multipath. While alternating among the individual paths of a k-short collection is preferred for the purpose of load balancing, using as many of them as possible in parallel is preferred for the purpose of robustness. Clearly, a possible choice is to use a subset of the k-short paths for parallel transmission of the packets, while alternating the elements of such subsets for load balancing. Another direction for future work is investigating whether, and to what extent, multiple (and/or disjoint) trees can be beneficial when certain nodes are processing multiple queries. In each of these settings, an important parameter that will need to be considered is the synchronization of the sleeping schedule of the nodes. Lastly, although we demonstrated that alternating among trees and multipaths, as well as using disjoint trees, can bring some lifetime-extensions, more work is needed in the direction of processing aggregated queries with such spatio-temporally varying routing structures.

7. REFERENCES

[1] Mica2 wireless measurement sheet, http://www.xbow.com/, crossbow technology inc., san jose, ca.

[2] K. Akkaya and M. Younis. A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3(3), 2005.

[3] P. Andreou, D. Yazti, P. Chrysanthis, and G. Samaras. Workload-aware query routing trees in wireless sensor networks. In *MDM*, 2008.

[4] P. Bose, E. D. Demaine, F. Hurtado, J. Iacono, S. Langerman, and P. Morin. Geodesic ham-sandwich cuts. In *Symposium on Computational Geometry*, 2004.

[5] N. Bulusu, J. D. Heidemann, D. Estrin, and T. Tran. Self-configuring localization systems: Design and experimental evaluation. *ACM Transactions on Embedded Computing Systems*, 3(1), 2004.

[6] C. Buragohain, S. Gandhi, J. Hershberger, and S. Suri. Contour approximation in sensor networks. In *DCOSS*, 2006.

[7] Q. Cao, T. Abdelzaher, T. He, and J. Stankovic. Towards optimal sleep scheduling in sensor networks for rare-event detection. In *IPSN*, 2005.

[8] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational geometry: algorithms and applications*. Springer-Verlag New York, Inc., 1997.

[9] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *Mobile Computing and*

Communications Review, 5(4), 2001.

[10] Y. Ganjali and A. Keshavarzian. Load balancing in ad hoc networks: Single-path routing vs. multi-path routing. In *INFOCOM*, 2004.

[11] M. T. Goodrich and R. Tamassia. Dynamic ray shooting and shortest paths in planar subdivisions via balanced geodesic triangulations. *J. Algorithms*, 23(1):51–73, 1997.

[12] A. Kaneko and M. Kano. Discrete geometry on red and blue points in the plane - a survey, 2003. http://gorogoro.cis.ibaraki.ac.jp/web/papers/kano2003-48.pdf.

[13] F. Kuhn, R. Wattenhofer, and A. Zollinger. An algorithmic approach to geographic routing in ad hoc and sensor networks. *IEEE/ACM Trans. Netw.*, 16(1):51–62, 2008.

[14] S. Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks. In *IEEE-ICC*, 2001.

[15] W. Luo, W. Liu, and Y. Zhang. Performance optimizatin using multipath routing in mobile ad hoc and wireless sensor networks. *Combinatorial Optimization in Communication Networks*, 2, 2005.

[16] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. Tag: a tiny aggregation service for ad hoc sensor network. In *Proc. Fifth Symp. on Operating Systems Design and Implementation, USENIX OSDI*, 2002.

[17] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. Tinydb: An acquisitional query processing system for sensor networks. *ACM TODS*, 30(1), 2005.

[18] A. Manjhi, S. Nath, and P. B. Gibbons. Tributaries and deltas: Efficient and robust aggregation in sensor network streams. In *SIGMOD Conference*, pages 287–298, 2005.

[19] D. Niculesu and B. Nath. Trajectory based forwarding and its applications. In *MOBICOM*, 2003.

[20] G. Parissidis, V. Lenders, M. May, and B. Plattner. Multi-path routing protocols in wireless mobile and ad hoc networks: A quantitative comparison. In *NEW2AN*, 2006.

[21] D. K. S. Bespamyatnikh and J. Snoeyink. Generalizing ham sandwich cuts to equitable subdivisions. *Discrete Computational Geometry*, 24, 2000.

[22] P. Thulasiraman, S. Ramasubramanian, and M. Krunz. Disjoint multipat routing to two distinct drains in a multi-drain sensor network. In *INFOCOM*, 2007.

[23] S. Wu and K. S. Candan. Power-aware single- and multipath geographic routing in sensor networks. *Ad Hoc Networks*, 5(7):974–997, 2007.

[24] Y. Yang, H.-H. Wu, and H.-H. Chen. SHORT: shortest hop routing tree for wireless sensor networks. *International Journal of Sensor Networks*, 2(5/6), 2007.

[25] V. Zadorozhny, P. Chrysanthis, and A. Labrinidis. Algebraic optimization of data delivery patterns in mobile sensor networks. In *MDDS Workshop*, 2004.

[26] W. Zhang and G. Cao. Optimizing tree reconifguration for mobile target tracking in sensor networks. In *IEEE INFOCOM*, 2004.

[27] W. Zhang, G. Cao, and T. L. Porta. Dynamic proxy tree-based data dissemination schemes for wireless sensor networks. *Wireless Networks*, 13(5):583–595, 2007.

[28] A. Zollinger. Geographic routing. In *Algorithms for Sensor and Ad Hoc Networks*, pages 161–185, 2007.