Load-Balancing Query Hotspots For Next-Generation Sensornets

Mohamed Aly
Google, Inc.
1600 Amphitheater Pkwy
Mountain View, CA 94043
Email: maly@google.com

Anandha Gopalan
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260
Email: axgopala@cs.pitt.edu

Adel Youssef
Google, Inc.
1600 Amphitheater Pkwy
Mountain View, CA 94043
Email: adel@google.com

***Abstract*—The expected architecture of next-generation sensornets raises the need for load-balanced point-to-point routing protocols to cope with different sources of traffic skewness. Query hotspots are one of these skewness sources that highly impact the sensornet performance. In this paper, we present a set of content-based load-balancing primitives to be used on top of any point-to-point routing protocol in order to detect and decompose query hotspots. Our schemes are based on local hotspot detection by the sensors targeted by queries. Hotspots are then decomposed by avoiding duplication when forwarding results to the query issuers. Our simulation results show the high benefit, in terms of network lifetime and throughput, of using our schemes to load-balance query hotspots of different sizes.**

I. Introduction

Most early sensornet deployments targeted data collection and push-based querying, e.g. [1], [2]. Hence, most current sensornet code bases, e.g. [3], [4], mainly offer tree-based many-to-one and one-to-many routing, query broadcasts, and aggregation during the data collection process [5], [6]. Many early sensornet applications were based on this model, e.g. monitoring and surveillance applications [7]. However, researchers and practitioners envision the next-generation sensornets to be composed of sensors deployed everywhere on the globe, together with gateways connecting sensors to Internet users/applications [8], [9]. Gateways may be stationary base stations [10], or mobile ones, such as robots, cell phones, and PDAs [11], [12]. In this model, querying loads will mostly be composed of pull-based ad-hoc queries issued by mobile users and/or Internet users. Ad-hoc queries trigger the need for using point-to-point routing [13], [14], a different routing paradigm from the old sensornet data collection model.

The large and continuously varying number of query sources in next-generation sensornets highly complicates the task of predicting the query distributions. Furthermore, the possibility of traffic skewness, when the sources and/or the destinations of most queries belong to a fairly small subset of sensors, is high. Query hotspots [15], where most queries access a fairly small number of nodes simultaneously, represent one of the major traffic skewness sources. Query hotspots may be in the form of Data-Centric Storage(DCS) range queries [15], e.g. many queries asking for a small range of temperature readings stored in one or two sensors, or geo-centric queries [8], when many users are simultaneously interested in data generated by sensors in a particular area (e.g. find free parking spots in downtown area). Traffic skewness, and in particular query hotspots, is a major problem that may result in the early death of sensors, network partitioning, and a subsequent reduction in network lifetime. Unfortunately, none of the currently available point-to-point routing schemes is equipped with a load-balancing functionality that decomposes query hotspots.

This paper presents three *content-based* schemes to detect and decompose query hotspots in next-generation sensornets. We assume the underlying sensornet implements a point-to-point routing scheme and is accessible by multiple base stations. Queries may either be DCS or geo-centric queries. A query is issued from a base station to a nearby sensor. This *query issuer* sensor is then responsible for forwarding the query to the *data source(s)*, the sensor(s) addressed by the query. Results are then forwarded to the issuer that, in its turn, answers the base station(s).

To decompose query hotspots, our schemes avoid duplication in forwarding results of similar queries. Our proposed solution is composed of two phases: *local hotspot detection*, and *hotspot decomposition*. The hotspot detection is solely determined by each data source sensor. At its high level, a sensor keeps track of the recent queries it answered (or currently answering), together with the issuer(s) of each of these queries. The sensor detects a possible hotspot when two (or more) queries intersect. Based on the nature of the intersection and the location of the issuers (physical or logical), the hotspot decomposition phase consists of one of three solutions: *two-phase query processing*, *three-phase query processing*, and *query partitioning*. We show through extensive simulations, that the major advantages achieved by applying our schemes on top of geographic routing protocols are:

- Load-balancing query hotspots and thus increasing the network lifetime and throughput.
- Maintaining a comparable level of Quality of Service (QoS) and real-time guarantees to that offered by the underlying routing protocol.

Paper Organization: Related work is presented in Section 2 and our solutions are presented in Section 3. Section 4 presents experimental results and Section 5 concludes the paper and discusses future work.

II. Related Work

In this section, we first provide a quick review of the load-balancing protocols already presented in literature. Then, we briefly classify the currently available point-to-point routing protocols.

Load-Balancing: Query hotspots have been first addressed in sensornets by Aly *et al.* [15]. The authors dealt with the problem when arising in DCS schemes and decomposed hotspots by moving hot data away from the hotspot area. Most of the other previously presented load-balancing paradigms were embedded in routing protocols. Many of these protocols were based on multipath routing, where a set of paths are determined for each packet type prior to the network operation and paths are interchangeably used afterwords [16], [17]. Directed diffusion [18] presented the idea of finding multiple routes from multiple sources to a single destination while applying in-network data aggregation. Many multipaths routing schemes were later presented based on Directed Diffusion, e.g. [19], [20]. To our knowledge, no content-based load-balancing schemes have been presented in literature.

Point-to-Point Routing: The need for point-to-point routing has recently increased as many current sensornet applications assume its usage, e.g. data-centric storage (DCS) [21], [22] and muti-dimensional range queries [23], [24], [15]. The first point-to-point routing schemes presented for wireless and sensor networks were based on *geographic routing*, e.g. GPSR [13], where nodes are identified by their geographic coordinates and routing is done greedily. Later, it was pointed that geographic schemes suffer from various limitations, e.g. the inability of current radios to conform with the current planarization algorithms and the unrealistic requirement of GPS-equipped sensors [14], [25]. Driven by these problems, schemes like NoGeo [26] and GEM [27] suggested the use of synthetic *virtual* coordinates assigned by iteratively embedding nodes in a Cartesian plane. Two recent schemes, BVR [14] and Logical Coordinates [28], used a collection of ideas from both geographic and virtual coordinates schemes. The basic idea of both schemes is to let nodes obtain coordinates from a set of landmarks. Routing then minimizes a distance function on these coordinates.

III. Load-Balancing Schemes

We now present our load-balancing schemes. Our proposed solution is based on a two-steps process. The first step is the local hotspot detection, and the second is the hotspot decomposition. Our hotspot detection scheme is a fully distributed scheme where each node maintains a list of the recent queries that it received together with the issuer node of each of these queries. Whenever a node receives a new query, it compares it with queries in the list. When detecting that the new query intersects, fully or partially, with one from the list, or when two concurrent queries intersect, the node applies one of the schemes described below.

The effectiveness of applying any of the load-balancing schemes below directly depends on the ability of each node to keep a reasonable history of recent queries it answered (or participated in answering). This is usually not a big problem as keeping track of recent queries does not consume much memory of the node's cache. However, all the load-balancing schemes described in this paper assume that each query issuer is able to temporarily cache the results of the query it issued. Recall that queries are originally generated by base stations. Thus, for stationary base stations, results can be stored in the base station probably having much more storage than surrounding sensors (i.e., query issuers). As for mobile base stations, they can access any of the sensors. Thus, assuming mobile base stations are likely to be nearby any of the sensors with an equal probability, caching the results of the recently issued queries would not cause any storage overload on any of the issuer sensor nodes.

For decomposing the query hotspot, we present three content-based hotspot decomposition schemes in the three subsections below.

A. Two-Phase Query Processing

Our first load-balancing scheme is based on detecting that two issuers are asking for the same data simultaneously or within a small time duration. The idea is to avoid duplication in sending results by answering one of the issuers and asking it to forward the results to the second issuer. The selection of the issuer to send results to depends on the approximate positions of the two issuers with respect to the data source.

We now illustrate the above scheme. When detecting two intersecting queries, the data source s should expect a gain to be achieved by applying the above scheme. This gain cannot be achieved if the data source is falling in between both issuers r_1 and r_2 as, in such case, sending results separately to each issuer would be less costly in terms of the collective energy consumed by all nodes involved in forwarding the results. The scheme would result in an energy gain when both r_1 and r_2 are falling in *one direction* with respect to s. By deducing that $length(\overline{sr_1}) + length(\overline{r_1r_2}) < length(\overline{sr_1}) + length(\overline{sr_1})$, where $length(\overline{ab})$ is the *approximate* length of the line between a and b either geographically or in terms of hops, s forwards the results to r_1, which is the nearer issuer. This results in a double gain. The first is to avoid overloading the data source (and the nodes around it) with the burden of answering many duplicate queries. The second is to balance the load among sensor nodes in the network.

B. Three-Phase Query Processing

When both r_1 and r_2 are on the same side with respect to s, the above scheme may fail to achieve an energy benefit if $length(\overline{r_1r_2})$ is almost equal to both $length(\overline{sr_i})$, for $i = 1, 2$ (e.g. nodes forming an equilateral triangle). We now present the three-phase query processing scheme where results are sent to an intermediate destination then forwarded to both issuers.

The scheme works as follows. When the data source determines that $length(\overline{r_1r_2}) \simeq length(\overline{sr_i})$, for $i = 1, 2$, it sends the intersecting results to a node t falling between r_1 and r_2. Determining t is done on a hop-by-hop basis by appending the packets carrying the query results with both final destinations r_1 and r_2 and forwarding these packets toward $bisector(\overline{r_1 r_2})$, the physical (or logical) bisector of the line $\overline{r_1 r_2}$. Upon receiving any such a packet p, a sensor c checks whether continuing to forward p toward $bisector(\overline{r_1 r_2})$ would move it closer to, or further from, its final destinations. For the second case, c stops the forwarding process and sends a copy of the packet to each of the destinations. The energy gain of this scheme is the most when the nodes s, r_1, and r_2 really fall on an equilateral triangle.

C. Query Partitioning

The previous two schemes considered intersections between simultaneous queries. Our third load-balancing scheme, namely query partitioning, considers load-balancing when a query intersects with previous queries answered by the same data source.

In general, two or more queries simultaneously addressing a data source may intersect among each other in part of their results, and at the same time, may intersect with one ore more queries that recently addressed the same data source. In such a case, the data source detects the two types of intersections. For the first type of inter-query intersection, the data source applies either the two-phase or the three-phase query processing technique depending on the locations of the query issuers (as described in the previous two sections). For the second intersection type, the data source redirects the intersecting part of the query to be answered by the most recent issuer of that part. This can be done through sending a single packet to this issuer, x. Then, x acts like a new data source receiving a new query and processing it using data cached in its memory. In case the query result is to be sent to more than one issuer of the original query, x applies either two-phase or three-phase query processing. Otherwise, the result is forwarded to the only issuer requesting it. Assuming that a query is usually composed of more than one packet, this scheme decreases the load on data sources and maximizes the benefit of caching query results.

IV. Experimental Results

To evaluate the performance of the proposed schemes, we implemented them on top of GPSR [13] using the Glomosim wireless network simulator [29]. We simulated a typical next-generation sensornet with *multiple base stations*, both stationary and mobile. Sensors are stationary and randomly distributed in a service area A. Sensors have an equal starting energy amount of $e = 100$ units. A sensor consumes 0.25 and 1 for receiving and sending one packet, respectively. Whenever a sensor s sends a packet p to a neighbor t, only s and t consume energy for sending and receiving p. The wireless medium is assumed to be reliable and does not contribute to any packet loss.

We ran simulations for networks of sizes varying between 1000 and 2000 sensors. The service area, A spanned a 200x200 square. At the start of every simulation, node locations are picked at random. Initially, each node broadcasts one message to know its neighbors' locations and it receives as many messages as the number of its direct neighbors. No maintenance messages are further sent during the simulation.

Query hotspots are generated as follows. At the start of each simulation, a small amount of stationary base stations are randomly selected. Few sensors are selected at random to be the hotspot centers. Queries of random result sizes, in terms of packets, are issued from base stations. The result of a query is a random selection of packet labels from a limited pool of numbers. If packets belonging to two different queries have the same label, then the two queries are said to intersect. A high percentage of the queries are sent from the stationary base stations to the selected destinations. The rest of the queries are sent from random sources to random destinations as to model queries issued by mobile nodes, picked by a nearby sensor, and targeting another sensor in the network.

We analyze the performance using the following three metrics: *network lifetime*, *throughput*, and *QoS*. We define the network lifetime to be the time elapsed before the death of the first node in the network. Throughput measures the number of successfully sent packets by all network nodes before the network dies. QoS is measured in terms of the time taken by each packet to reach its destination. Network lifetime gives an idea of how the proposed schemes load-balance the energy consumption among the different sensor nodes. Throughput on the other hand shows how load-balancing query hotspots increases the network performance in terms of the number of successfully sent packets. QoS measures the delay overhead that the schemes add compared to plain GPSR.

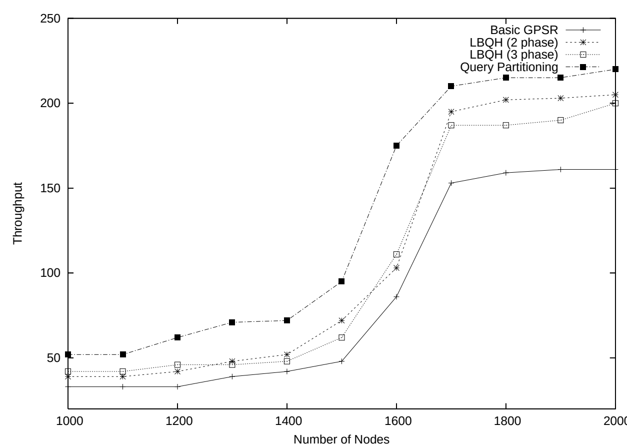

Fig. 1. The effect of load-balancing schemes on throughput

Fig. 1 compares applying each of our schemes separately to the network with plain GPSR in terms of throughput. The figure shows that applying either two-phase or three-phase query processing achieves throughput improvement independently from the network size. It should be noted that two-phase slightly outperforms three-phase for large networks as the first deals with more frequently occurring cases. The figure also shows that applying query partitioning significantly increases throughput. The performance increases as the network size increases. This shows the benefit of dealing with all types of query intersections in the query partitioning scheme.

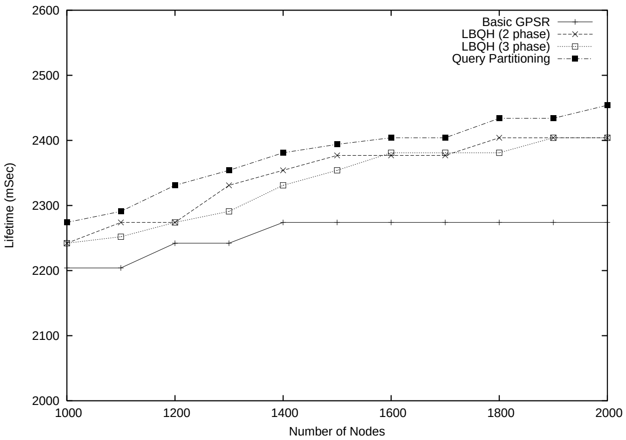

Fig. 2. The effect of load-balancing schemes on network lifetime

Fig. 2 studies the effect of the proposed load-balancing schemes on network lifetime. As in Fig. 1, applying two and three phase query processing outperforms plain GPSR, while applying query partitioning outperforms the first two schemes. The gain increases with the increase in the network size. It should be noted that the constant network lifetime for GPSR between points 1400 and 2000 is due to the deterministic behavior of GPSR against the same type of skewness.

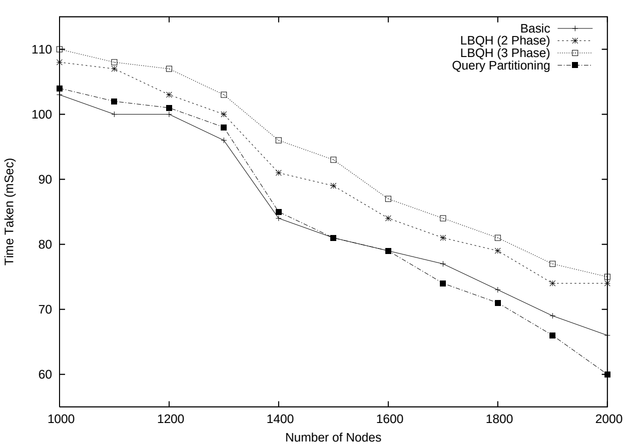

Fig. 3. Quality of Service(QoS)

Fig. 3 checks the QoS improvement of the proposed schemes in terms of the average time taken by the query to be answered. The figure shows that all schemes have comparable QoS as differences are in terms of milliseconds. However, the figure shows that two-phase and three-phase query processing achieve the worst QoS compared with plain GPSR. This is because, in both schemes, some packets may take a detour before reaching their originally intended destination. The figure also shows that query partitioning has almost similar QoS compared to plain GPSR for small to medium network sizes and slightly outperforms GPSR for networks larger than 1600 nodes. This shows the benefit of considering intersections with past queries.

Fig. 4 and 5 show how the two and three phase schemes

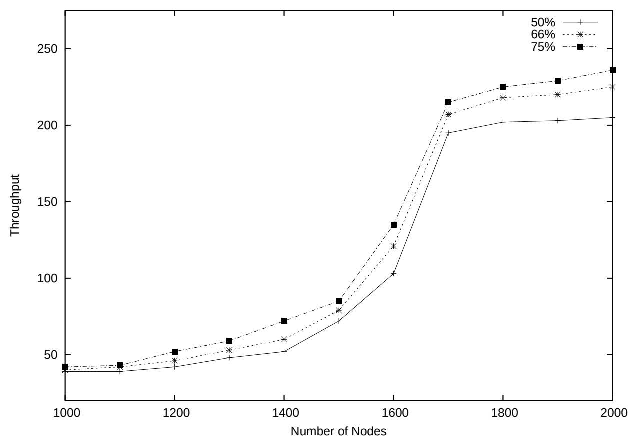

Fig. 4. The effect of varying intersection level on the two-phase scheme

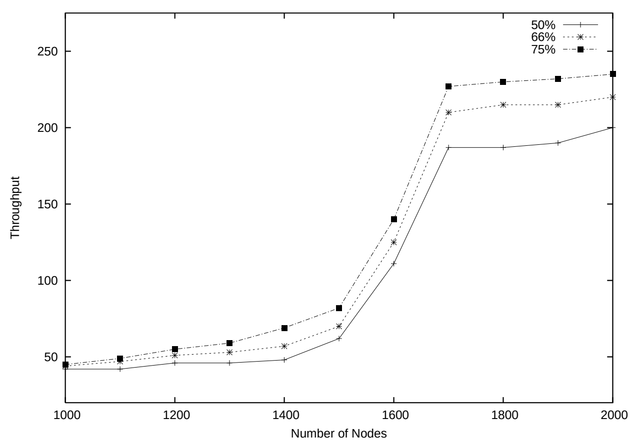

Fig. 5. The effect of varying intersection level on the three-phase scheme

perform when we vary the percentage of intersections between simultaneous queries. For intersections of 50%, 66%, and 75%, the two figures show that the higher the intersection the better the achieved throughput by both schemes. The throughput improvement of each scheme increases with increasing the network size. This gap is slightly larger for the three-phase scheme than two-phase. This may be justified by the fact that for higher intersections and large networks, the amount of avoided duplication is higher in the three-phase scheme.

Fig. 6 shows how the query partitioning scheme performs when we vary the percentage of intersections between current queries and previous queries. The figure shows a similar performance to that achieved for the other two schemes. However, the throughput gain between different intersection levels for large networks is larger than the similar gain for the previous two schemes. The reason behind this is that exploiting the intersection between current and past queries significantly balances the load among sensors.

V. Conclusions and Future Work

In this paper, we presented a set of content-based load-balancing schemes to be run on top of point-to-point routing schemes in order to decompose query hotspots in next-

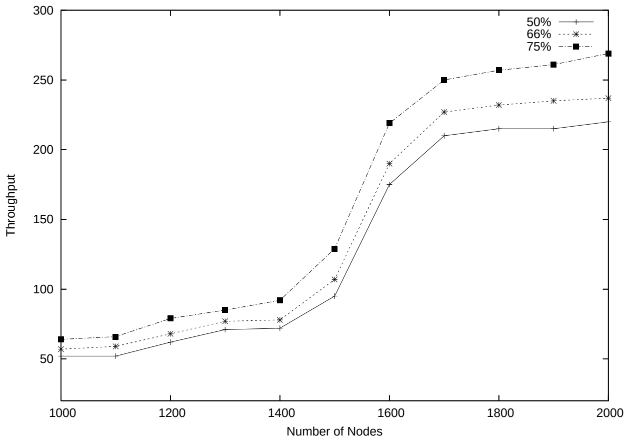

Fig. 6. The effect of varying intersection level on the query partitioning scheme

generation sensornets. Our schemes are based on locally detecting query hotspots and avoiding duplicate query answering by detecting intersections among simultaneous and past queries. Experimental evaluation showed the benefit achieved by our schemes, in terms of increasing throughput and network lifetime, when compared to plain point-to-point routing. In the future, we plan to implement our schemes on current sensornet testbeds to physically test their performance for various network settings.

REFERENCES

[1] P. Bonnet, J. Gehrke, and P. Seshadri, "Towards sensor database systems," in *Proc. MDM*, 2001.

[2] Y. Yao and J. Gehrke, "Query processing for sensor networks," in *Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR 2003)*, 2003.

[3] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," in *Proc. of ASPLOS*, 2000.

[4] L. Girod, T. Stathopoulos, N. Ramanathan, J. Elson, D. Estrin, E. Osterweil, and T. Schoellhammer, "A system for simulation, emulation, and deployment of heterogeneous sensor networks," in *Proc. of SenSys*, 2004.

[5] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," vol. 36, no. SI. New York, NY, USA: ACM Press, 2002, pp. 131–146.

[6] H. Gupta, V. Navda, S. R. Das, and V. Chowdhary, "Efficient gathering of correlated data in sensor networks," in *Proc. of MobiHoc*, 2005.

[7] J. Zhao, R. Govindan, and D. Estrin, "Sensor Network Tomography: monitoring wireless sensor networks," *Computer Communication Review 32(1)*, vol. 64, 2002.

[8] S. Nath, J. Liu, J. Miller, F. Zhao, and A. Santanche, "Sensormap: a web site for sensors world-wide," in *Proc. of SenSys*, 2006.

[9] T. Luckenbach, P. Gober, S. Arbanowski, A. Kotsopoulos, and K. Kim, "Tinyrest - a protocol for integrating sensor networks into the internet," in *Proc. of REALWSN*, 2005.

[10] H. Dai and R. Han, "Unifying micro sensor networks with the internet via overlay networking," in *Proc. of LCN*, 2004.

[11] C. Westphal, "Scaling properties of routing protocols in sensor networks with mobile access," Nokia, Tech. Rep., July 2006.

[12] Z. Vincze, D. Vass, R. Vida, A. Vidacs, and A. Telcs, "Adaptive sink mobility in event-driven multi-hop wireless sensor networks," in *Proc. of InterSense*, 2006.

[13] B. Karp and H. T. Kung, "GPSR: Greedy perimeter stateless routing for wireless sensor networks," in *Proc. of ACM Mobicom*, 2000.

[14] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, and I. Stoica, "Beacon Vector Routing: Scalable point-to-point routing in wireless sensornets," in *Proc. of NSDI*, 2005.

[15] M. Aly, P. K. Chrysanthis, and K. Pruhs, "Decomposing data-centric storage query hot-spots in sensor networks," in *Proc. of MOBIQUITOUS*, 2006.

[16] R. C. Shah and J. M. Rabaey, "Energy aware routing for low energy ad hoc sensor networks," in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, 2002.

[17] S. Dulman, T. Nieberg, J. Wu, and P. Havinga, "Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks," in *Proc. of WCNC*, 2003.

[18] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, , and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Transactions on Networking (TON)*, vol. 11, February 2003.

[19] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wireless sensor networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, 2001.

[20] I. Raicu, L. Schwiebert, S. Fowler, and S. K. Gupta, "Local load balancing for globally efficient routing in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 1, 2005.

[21] S. Shenker, S. Ratnasamy, B. Karp, R. Govidan, and D. Estrin, "Data-centric storage in sensornets," in *Proc. of HotNets-I*, 2002.

[22] M. Aly, K. Pruhs, and P. K. Chrysanthis, "KDDCS: A load-balanced in-network data-centric storage scheme in sensor network," in *Proc. of CIKM*, 2006.

[23] X. Li, Y. J. Kim, R. Govidan, and W. Hong, "Multi-dimensional range queries in sensor networks," in *Proc. of ACM SenSys*, 2003.

[24] M. Aly, N. Morsillo, P. K. Chrysanthis, and K. Pruhs, "Zone Sharing: A hot-spots decomposition scheme for data-centric storage in sensor networks," in *Proc. of DMSN*, 2005.

[25] Y.-J. Kim, R. Govidan, B. Karp, and S. Shenker, "On the pitfalls of geographic face routing," in *Proc. of DIALM-POMC*, 2005.

[26] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica, "Geographic routing without location information," in *Proc. of Mobicom*, 2003.

[27] J. Newsome and D. Song, "GEM: Graph embedding for routing and data centric storage in sensor networks without geographic information," in *Proc. of SenSys*, 2003.

[28] Q. Cao and T. Abdelzaher, "A scalable logical coordinates framework for routing in wireless sensor networks," in *Proc. of RTSS*, 2004.

[29] L. Bajaj, M. Takai, R. Ahuja, R. Bagrodia and M. Gerla, "Glomosim: A scalable network simulation environment," UCLA, Tech. Rep. 990027, May, 1999.