A Multipath Routing Algorithm for Mobile Wireless Sensor Networks

Alexander Aronsky and Adrian Segall
Department of Electrical Engineering, Technion IIT, Haifa, Israel
aronsky@tx.technion.ac.il, segall@ee.technion.ac.il

Abstract—**Providing reliable and yet energy efficient routing protocols is of an utmost importance in Wireless Sensor Networks. Wireless Sensor Networks imply multi-hop data forwarding over unreliable and moving nodes. When selecting the appropriate routing paradigm, the main challenge is to find the right equilibrium between the quality of data delivery and the energy invested. Insufficient quality of data delivery may fail the application deployed over the wireless sensor network, while an energy wasteful protocol may significantly shorten the lifetime of the network, thus making the deployment inefficient for its purpose. In this paper we propose a new routing algorithm for WSN's, named the Data Centric Braided Multipath (DCBM) algorithm. The algorithm is designed to achieve and maintain route resiliency through multiple interleaving routing paths, capable to cope with node mobility. Simulations show that the algorithm maintains a delivery ratio similar to the previously suggested protocols, but requires significantly lower control packet overhead. The algorithm has several additional desirable properties, like loop reduction and localized path repair.**

I. INTRODUCTION

Data dissemination and routing is a key issue in deploying Wireless Sensor Networks. The goal is to provide reliable data delivery from multiple sensors to collector nodes (sinks). One of the main issues that differentiate this routing environment from classical networks is the energy limitation of the participating nodes. Therefore tension exits between the control packets overhead invested in maintaining reliable data delivery and the lifetime of the deployed network. The balance between the reliability and energy investment had been previously investigated in MANETs (Mobile ad-hoc networks). MANET algorithms, like AODV[1], DSR[2] and DSDV[3] were found adequate for handling the challenge. But previous works, such as [4],[5] and [6], show that these algorithms are not well suited to cope with the distinctive WSNs characteristics.

The WSN main distinctive attributes are multiple collaborative sensing sources communicating with a single destination and reduced importance of topological changes that occur in areas other than that of the sensed phenomena. These features lead to the introduction of several new key paradigms. The *Data Aggregation* paradigm concentrates on methods to aggregate sensed data locally and limit the amount of information propagated through the network. The *Data Centric* routing paradigm focuses on effective establishment and maintenance of routes between the multiple data sources and a single sink. *Data Aggregation* typically implies hierarchy and clusters. However the complexity of establishing and maintaining clusters makes it worthwhile only in the static WSNs. In this paper, we concentrate on the non-hierarchical approach. When examining the non-hierarchical protocols, two types of protocols [7] emerge. They are both *Data Centric*, namely route establishment is triggered by a query sent by a sink and route maintenance takes into account the quality of data arriving at the sink. In the *Reverse-path-based forwarding* approach, data reports flow towards the sink, namely in the direction opposite to the query propagation. The sink sends out query messages that express its interest, normally using flooding. Whenever a node receives a query from a neighbor, it sets up a forwarding state in the form of a pointer from itself to the neighbor, thus indicating the reverse data path. Data reports generated by sources travel along the pointers from one node to the next, until they reach the sink. Otherwise seen, a *Reverse-path-based forwarding* is a *Distance Vector* algorithm with the vector dimension of 1, since there is only one destination of the data, the *sink*. Examples of this approach are Directed Diffusion (DD)[8] and Directed Diffusion with Stepwise Interest Retransmission (DDSIR)[9]. In the second approach, referred to as *Cost field-driven dissemination*, the forwarding states of the nodes consist only of the cost denoting the distance to the sink, measured in certain units, like hop count, expected energy consumption or physical distance. The cost value is directionless, but implies direction, in the sense that data from each node can flow only to neighbors with smaller cost. Examples of this approach are Gradient Broadcast (GRAB)[10] and Reliable Cost-based Data-centric Routing (RCDR) [11] . It is worth pointing out here that, while GRAB and DD address only environments with static nodes, DDSIR and RCDR consider both static and mobile node environments. Node mobility is also addressed in [14] and [15]. The authors of [14] consider sink mobility, which is an interesting issue not considered in the present version of our protocol DCBM and is an aspect that might be of interest in future work. In [15], the authors develop a metric that incorporates prediction of link availability, transmission energy and residual energy. This and other metrics, like ETX [16], might be used in future work to possibly improve the performance of our and other routing protocols. In the present paper we use the metric of residual energy, which has been shown in [17] to be the most effective, compared with other metrics, in terms of energy consumption in systems with uniform transmission power.

Our protocol, named Data Centric Braided Multipath (DCBM), is a *Reverse-path-based forwarding* algorithm. As in all Distance Vector algorithms, there is always the balance between using previous distance information when the network operates normally, and disregarding this information in cases when topological changes make it irrelevant. The latter is most prominent for example when a routing loop occurs. DCBM achieves good performance in terms of high data delivery rate and low overhead, by:

- Normally using previous information, but detecting loops and dismantling them soon after their formation
- Monitoring data delivery and limiting refreshes to the neighborhood of paths to sources with poor delivery quality
- Maintaining braided multiple paths from sources to the sink

In the sequel we shall describe the algorithm and indicate its main properties. Proofs of its properties and the algorithm pseudo-code appear in [12].

II. DCBM - PROTOCOL DESIGN

A. Path establishment

The primary goal of DCBM is to provide resilient and energy efficient multipath routing between sensor nodes and a sink, while minimizing control message overhead.

Path establishment and maintenance is performed by two-phase cycles initiated by the sink. In the first phase of a given cycle, control messages MSG1 are used to broadcast data queries, to carry metric data and to trigger the selection of designated *neighbors* for this cycle – the neighbor with the best known distance to the sink. The *designated neighbor* is the candidate to be the next hop in the reverse routing path. The first phase has two variants: *Fast Propagation Algorithm* and *Delayed Propagation Algorithm*.

The motivation for *Fast Propagation Algorithm* stems from sensitive applications requiring continuous data delivery even in case of significant topology changes. In this case, the objective is to reestablish data delivery in the shortest time frame possible. Another incentive is urgent changes in parameters related to monitoring or reporting in the deployed application, which may require fast query/interest diffusion. To meet these requirements, the *Fast Propagation Algorithm* forwards MSG1 messages in the fastest possible way, possibly at the expense of path energy and reliability considerations. In this version, nodes propagate MSG1 of a given cycle immediately or almost immediately after receiving the first MSG1 of this cycle. A short random delay may be introduced to minimize collisions between forwarded MSG1's.

The motivation for the *Delayed Propagation Algorithm* version is the attempt to achieve the longest possible lifetime of the wireless sensor network. This version attempts to minimize energy depletion across the WSN by delaying the rebroadcast of MSG1 by a time period proportional to the estimated distance to the sink[10]. The delay allows accumulation of neighbor information, thereby providing the opportunity to select a better *designated neighbor.*

In the second phase of the DCBM algorithm, control messages MSG2 are propagated on the paths formed by the *designated neighbors*. MSG2 is employed for the activation of the reverse routing paths established in the first phase and as a mechanism to prevent routing loops. MSG2 is rebroadcasted only if received from the *designated neighbor* and a node is allowed to forward data via any neighbor it has received a MSG2 from, provided that the cost-to-sink advertised by that node is less than the cost of the node itself. All those neighbors are included in the *eligible neighbor list.* The neighbor to which data is forwarded is selected for each packet from the *eligible neighbor list* and is referred to as the *active next hop*. This mechanism allows the establishment and maintenance of braided multi-paths for all sensor nodes..

B. Data Forwarding

Data packets are forwarded via the node with the lowest distance to the sink among the nodes in the *eligible neighbor list*. Each node caches the last forwarded data packet. If the data packet is not overheard (or acknowledged, depending on the system link layer) from the *active next hop* node after a given number of retransmission attempts, the latter is made invalid for forwarding purposes and is removed from the *eligible neighbor list.* If the node has another neighbor in the *eligible neighbor list*, the data packet is forwarded again. If no node is available, the packet is dropped and a prune control message is broadcast in order to prevent neighbors from using this node as next hop.

C. Route Maintenance

Local route integrity is maintained by the overhearing (or acknowledgement) mechanism mentioned in the previous section. To reiterate, each node manages an *eligible neighbor list.* A neighbor k is deleted from the list of node i if the data packet sent by i via k is not overheard as forwarded further by k. If the *eligible neighbor list* becomes empty, node i broadcasts a prune message.

Global route integrity is maintained by the sink. In addition to scheduled periodic path refreshes, the sink constantly monitors the quality of data delivery. The sink is assumed to be aware of all sources and of the expected data rate from each source. If the received data rate from a source drops below some threshold, the sink assumes that the topology has changed and no alternative path was found and triggers a new path establishment cycle. If the data delivery from more than a single source is disrupted, the sink triggers a global refresh cycle.

In mobile environments, we have observed a significant improvement of the *Fast Propagation Algorithm* if the sink triggers two consecutive cycles every time a refresh is called for. The second cycle helps nodes to avoid selecting neighbors that have in fact moved out of range as *designated neighbors*.

D. Localized Path Refresh

In order to reduce control overhead, it is important to localize refreshes of the routing paths if topological changes affect only the data flow from a single source. Localization is achieved by limiting participation in the refresh only to nodes that are close to the disrupted path. The limiting technique is implemented by the use of several lists and parameters, maintained at nodes and/or included in the control messages.

Each node maintains a list of sources, named *Active Source List*, whose data packets it is forwarding in the current cycle. Each control message MSG1 contains an additional triplet named *Source, ttl* and *TTL. Source* denotes the identity of the source whose data delivery has been disrupted, *ttl* is the distance the MSG1 is still allowed to travel from the current node on and *TTL* is the initial *ttl*.

To limit the flooding of the MSG1 control messages, the *ttl* value is decremented if the rebroadcasting node has not been an intermediate node on the disrupted path between *Source* and the sink (does not have *Source* in its *Active Source List*). Otherwise, the *ttl* parameter is reset by the node as *TTL*. If the value of *ttl* in a received MSG1 is 0, the message is discarded.

Global flooding is identified in MSG1 messages by setting the *Source* field to '-1'.

E. *Loop Handling*

Designated neighbor loops may occur due to non-updated information at the nodes. In this case, the nodes in the loop will not receive MSG2 from their designated neighbor and thus will not send MSG2 in the current cycle. As discussed below, DBCM detects and dismantles those loops as soon as possible, but while in the loop, nodes may still be partially active in receiving and forwarding data.

- A node that receives MSG2 from neighbors other than the *designated neighbor* can forward data via these neighbors, provided they are in the *eligible neighbor list*.
- The *eligible neighbor list* is invalidated only upon receipt of the first MSG2 of the cycle. Thus nodes in the loop may be in the *eligible neighbor list* compiled in the previous cycle by neighbors that also do not receive MSG2 in the current cycle. In this case, they may receive data messages from those neighbors. Thus when using the *Fast Propagation Algorithm* for forwarding urgent queries, we will not destroy previous paths without creating at least one new path.

If not taken care of, designated neighbor loops might stay in the network for a prolonged time. We use the following technique to speed up loop dismantling. Suppose a node does not receive MSG2 from its *designated neighbor* during several consecutive cycles. Then it suspects a designated neighbor loop, discards any previous distance information and, in the next refresh cycle, it selects the first neighbor it receives MSG1 from as the *designated neighbor*. This way the loop will be dismantled, since nodes use only new information. In order to force all neighbors that have selected this node as designated neighbor to also discard previous distance information, the MSG1 sent out by the node is a "poisoned MSG1".

III. PROPERTIES OF THE PROTOCOL

The properties of DCBM are given below. Proofs appear in [12].

- In each cycle, every node i sends at most one control packet MSG1 and at most one MSG2.
- Denote by $r_i[t]$ the active next hop of node i at time t and by $K(t) = \{(i, r_i[t]), \forall i\}$ connectivity graph of the network. There are no loops in $K(t)$ for any given time t.
- Suppose that changes in the network topology cease before the time when cycle c' starts (nodes are stationary, link weights are constant and propagation time is constant). Then a finite number of path refresh cycles afterwards, the distance parameter $D1_i[c]$ held by each node does not change and is identical to the optimal distance to the sink. In addition, the *designated neighbor* $e_i(c)$ is the next hop on the optimal path from *i* to sink.
- Let N be the number of nodes in the network and $t^{\max}$ the maximum propagation time between each two neighbors. If the time between two refresh cycles is larger than $3*N*t^{\max}$, then each Data Packet can be transmitted by a given node at most twice.

IV. SIMULATION RESULTS

A. *Simulation Environment*

We have used the ns-2 simulation environment [13] to simulate DCBM, and for comparison, also RCDR, GRAB, DDSIR and AODV. In this section, we shall list the simulation parameters and characteristics.

Node movement is a major characteristic of our environment. The Waypoint algorithm (part of ns-2) has been used to create simulation scenarios with node movement. The scenario generation algorithm sets random initial positions for all nodes in the network (uniform distribution). Then each node receives a randomly generated next interim location (uniform distribution) and movement speed (uniformly distributed between 0 and a *maximum speed*). Upon arrival to the interim location, the node briefly stays there and then a new pair is generated. We have varied the *maximum speed* parameter between 1*m/sec* and 5*m/sec*. The simulation field size is [1000 m x 1000 m]. The number of sensors distributed in the field change between 70 and 130. The transmission radius is 175 m, resulting in an average number of nodes in the transmission radius that varies between ~3.4 and ~6.25.

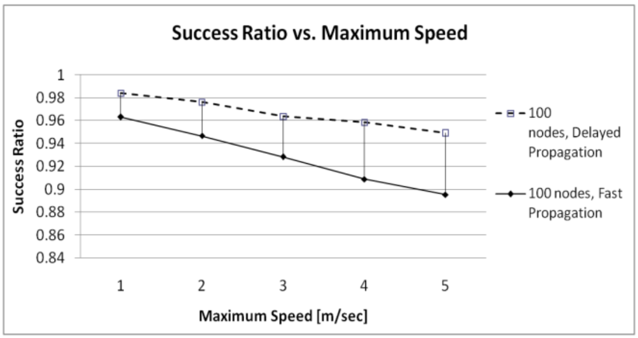
Each node that is sensing the phenomena to be reported to the *sink*, named *data source*, has constant data generation rate. The location of the sink and of the data sources is constant throughout the simulation scenarios. Each of those nodes is distanced 200 *m* from each border.

To evaluate the performance of the algorithms, we measure two parameters:

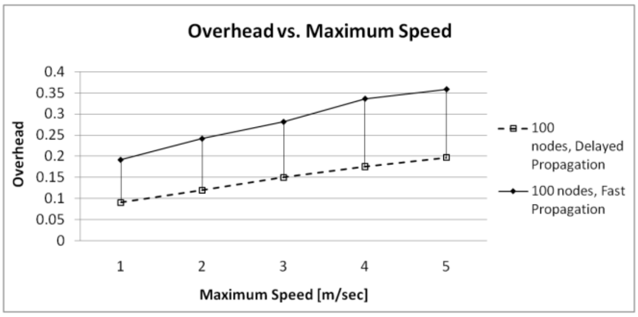
- Success ratio – percentage of data packets successfully delivered to the sink.
- Overhead – percentage of packets sent in excess of packets used to deliver data, i.e. number of duplicated data packets plus the number of control packets divided by the total number of transmitted packets.

B. *Comparison of the Algorithm Versions*

As said before, the performance of the *Fast* Propagation *algorithm* is greatly improved if the sink triggers two consecutive cycles every time refresh is called for. We have tried more than two consecutive cycles, but have observed no significant improvement. The second consecutive cycle is sufficient to ensure that neighbors that have moved out of the transmission range will not



a). Success Ratio vs. Maximum Speed



b). Overhead vs. Maximum Speed

Figure 1. Comparison of the Fast Propagation and the Delayed Propagation Algorithm

be selected as the *designated neighbor*.

As shown in Fig. 1.a) and 1.b) , the *Fast Propagation algorithm* provides smaller success ratio and larger control overhead than the *Delayed Propagation algorithm*. Fig. 1 also shows that the performance gap in both parameters increases with node mobility.

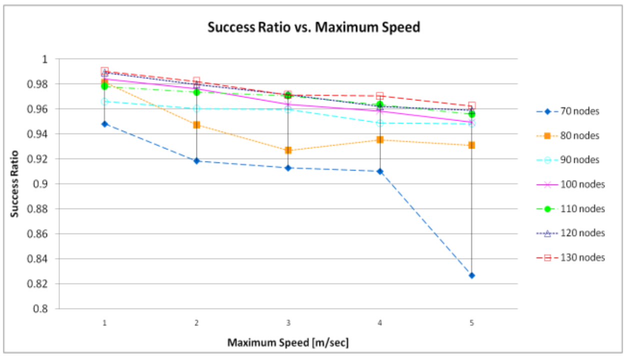
There are two main reasons for the performance gap, the first being the number of refresh cycles generated upon detection of a network topology change. The second reason is the frequency of *designated neighbor* loops. Each loop can significantly limit the propagation of MSG2. A limited propagation of MSG2 decreases the redundancy of the forwarding paths, thus creating more instances of packet loss and repeated refresh cycles. Another reason behind the performance gap is the fact that *designated neighbor*s are selected in the first version using fastest propagated, and thus less updated, MSG1's. Again the non optimal election of the *designated neighbor* reduces the number of nodes in the *eligible neighbor list* and decreases the redundancy of the forwarding paths.

Since the *Delayed Propagation Algorithm* seems to perform better, we shall use it in the rest of this chapter.

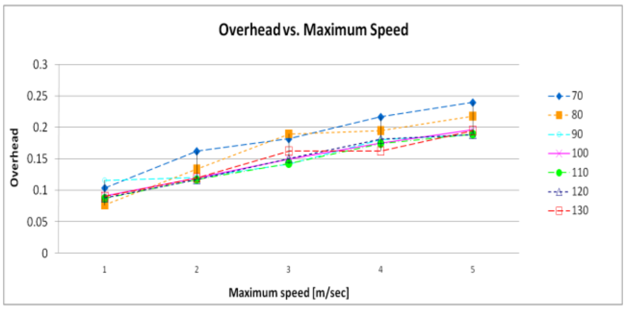
C. *Behavior of the Delayed Propagation Algorithm*

The results for various values of maximum node speed and node density are shown in Fig.2.

As expected, we see in Fig. 2.a) that the success ratio drops drastically in networks with low node density and high node mobility. Low node density results in a very small number of alternative paths and packets are frequently dropped. The sink detects decrease in the quality of delivered data and therefore new refresh cycles are generated often, which in turn increases the amount of control overhead. This is shown in Fig. 2.b).



a). Success Ratio vs. Maximum Speed



b). Control Overhead vs. Maximum Speed

Figure 2. Behavior of the Delayed Propagation Algorithm

We can also see that the overhead decreases significantly when the node density increases. High node density allows more redundant paths, therefore decreasing the number of required refresh cycles.

D. *Effect of limited refresh*

The purpose of the *Limited Refresh* enhancement is to limit the control overhead of the refresh cycles. We explore here the advantage of *Limited Refresh* and try to determine its optimal width, determined by the TTL parameter. In order to emphasize the effect, we alter the simulation environment as follows. The size of the simulation field is 1500m by 1500m and the number of nodes is 260. The sink and the data sources are positioned as in Fig. 3.

The enhancement dictates that if the sink detects deterioration of data rate from one source only, it performs *Limited Refresh*, but if the deteriorations are from more than one source, it performs *Global Refresh*.

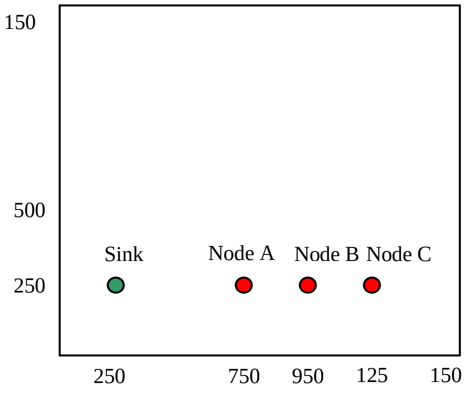
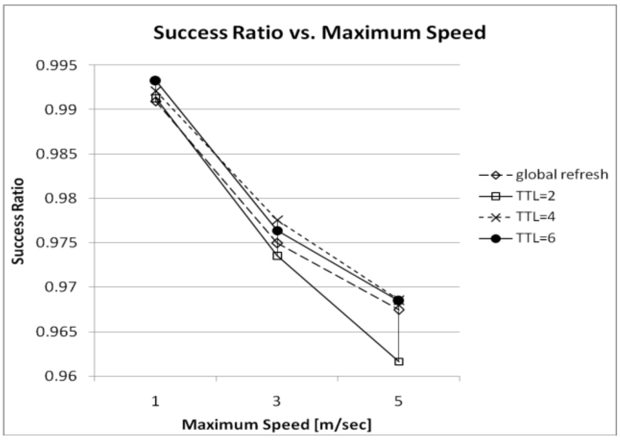
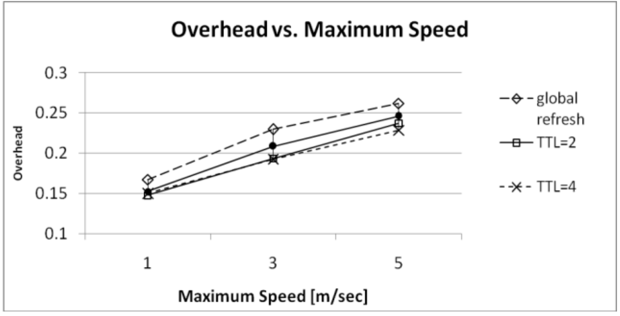


Figure 3. Sink and active nodes positioning



a). Success Ratio vs. Maximum speed



b). Overhead vs. Maximum speed

Figure 4. Limited Refresh performance

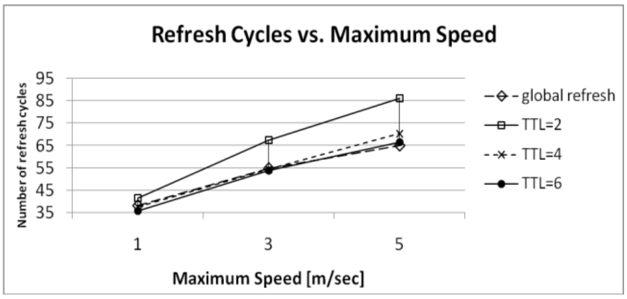


Figure 5. Limited refresh cycles

The purpose of the scenario is to show a case when all sources are concentrated in one direction. This is the case when Limited Refresh is most efficient, since it saves a large amount of overhead, while the success ratio is almost unaffected, as seen from Fig. 4. We also investigate how the TTL parameter affects the performance. TTL = 2 saves in overhead per cycle, but results in more cycles due to a smaller amount of alternative paths. TTL = 6 has the opposite effect and behaves almost as a global refresh. It seems that TTL=4 is a reasonable choice.

E. Comparison of algorithm performance

In addition to DCBM, we have used ns-2 to simulate several other protocols and to compare their performance to that of DCBM. In addition to the Sensor Network oriented algorithms, we have also included AODV, which we know already that does not apply well to these networks. Fig.6 displays the delivery success ratios of the algorithms.

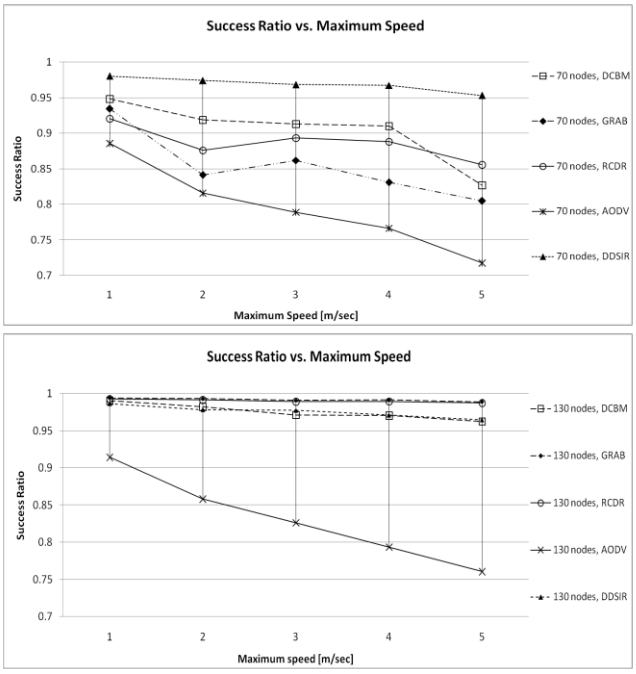


Figure 6. Algorithm comparison, Success Ratio vs. Maximum speed

We observe that low sensor density scenarios favor *Reverse-path-based forwarding* algorithms, like DDSIR and DCBM, over *Cost field-driven dissemination* algorithms, like GRAB and RCDR.

The reason is that in the latter, the forwarding mesh required for data delivery redundancy is very limited. DD/SIR performs better than DCBM because it does not require path redundancy. The data is transferred immediately upon receipt of the polling control message, before changes in topology may occur. RCDR performs better than GRAB because it has mechanisms to cope with node mobility and adapts the cost field by employing local neighbors' interactions. In high density scenarios, all algorithms perform similarly well. The conclusion is that DCBM performs overall very well, with only a small deterioration compared to the best one. It seems however that it is a price worth paying, given the big gain in terms of overhead, as seen from Fig. 7.

The overhead of *Reverse-path-based forwarding* algorithms is lower than the overhead of the *Cost field-driven dissemination* algorithms. This is due to the fact that the latter employs duplicate packets. RCDR has more overhead than GRAB because of the local cost field adaptation mechanism, which requires neighbor negotiation upon detecting a change in the neighbor status. DCBM has significantly less overhead than all the other algorithms, especially in high density networks. In particular, its overhead is much lower than that of DD/SIR, because the latter creates multiple polling cycles instead of a single one as used in DCBM.

CONCLUSIONS

We have shown that our *Reverse-path-based forwarding* algorithm, DCBM, is well suited to cope with mobile WSN environments. The main limitation of the WSN environment is the energy of the deployed sensors. DCBM creates and maintains a braided multipath forwarding scheme, whose maintenance requires a

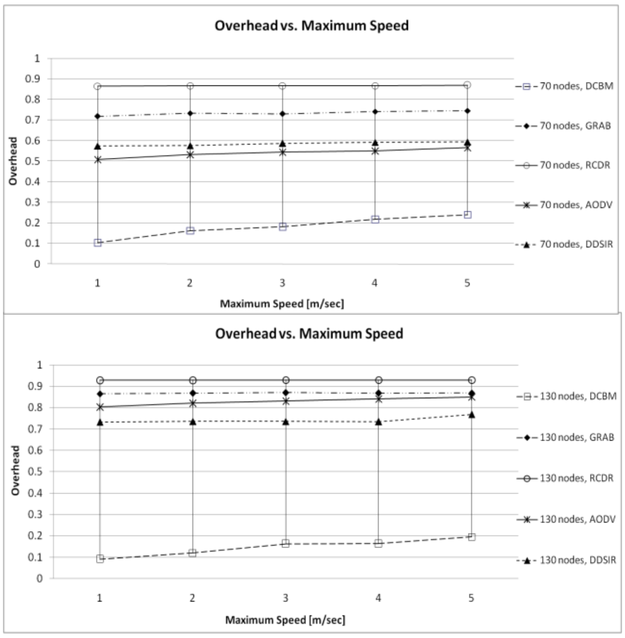


Figure 7. Algorithm comparison, Overhead vs. Maximum Speed.

relatively small amount of overhead. Furthermore, the redundancy of the braided multipath and the local maintenance mechanism allow a high level of success ratio in data delivery. We stated the properties of the algorithm, such as convergence to optimal path and loop avoidance. These properties are important when considering the deployment of the algorithm in real environments.

Our simulations suggest that the algorithm may be used for applications requiring a constant rate from data sources, like sensors that detect certain phenomena and are deployed in environments with sensor mobility. Examples of such applications can be the gathering of health information from tags deployed in livestock management systems, micro sensors deployed into patient blood streams and environmental monitoring, such as ocean stream monitoring.

REFERENCES

[1] C. E. Perkins and E. M. Royer. “Ad-hoc On-Demand Distance Vector Routing”. Proceedings of the 2-nd IEEE Workshop on Mobile Computing Systems and Applications, pages 90–100, New Orleans, LA, February 1999.

[2] D. Johnson, Y. Hu and D. Maltz. “The dynamic Source Routing Protocol for Mobile Ad Hoc Networks”, IETF Internet draft, Apr 2003.

[3] C. Perkins, “Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers” , ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications. Volume 24, Issue 4, Pages: 234 – 244.

[4] R. V. Boppana and A. Mathur, “Analysis of the Dynamic Source Routing Protocol for Ad Hoc Networks,” IEEE Workshop on Next Generation Wireless Networks (WoNGeN), December 2005.

[5] S. Dulman, J. Wu, P. Havinga, "An energy efficient multipath routing algorithm for wireless sensor networks", IEEE International Symposium on Autonomous Decentralized Systems (ISADS 2003), Pisa, Italy.

[6] T. Camilo, J. Sá Silva, F. Boavida, “Assessing the use of ad-hoc routing protocols in Mobile Wireless Sensor Networks” in Proc. of Conference on Mobile and Ubiquitous Systems, CMUS2006, Braga, June 2006.

[7] F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang, “Dissemination protocols for large sensor networks. Wireless sensor networks book contents”, in C.S. Raghavendra & al., Wireless Sensor Networks, Kluwer, ISBN:1-4020-7883-8, 2004, Pages: 109 - 128

[8] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. “Directed Diffusion for Wireless Sensor Networking” ACM/IEEE Transactions on Networking, Volume 11, Issue 1, pp. 2-16, February, 2002.

[9] M. Bandai, T. Mioki, and T. Watanabe, “A Routing Protocol with Stepwise Interest Retransmission for Wireless Sensor Networks”, IEICE TRANS. COMMUN., VOL.E91–B, NO.5 MAY 2008.

[10] F. Ye, G. Zhong, S. Lu, and L. Zhang, "GRAB: A Robust Data Delivery Protocol for Large Scale Sensor Networks," presented at IEEE IPSN, 2003.

[11] J. Wu, P. Havinga, “Reliable Cost-based Data-centric Routing Protocol for Wireless Sensor Networks” Seventh ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, SNPD 2006. June 2006, pp 267 – 272.

[12] A. Aronsky and A. Segall, Routing Protocols for Wireless Sensor Networks, CCIT Report 741, Technion, August 2009, http://webee.technion.ac.il/publication-link/index/id/539

[13] http://en.wikipedia.org/wiki/Ns-2

[14] Munari, A. Schott, W. Krishnan, S. "Energy-efficient routing in mobile wireless sensor networks using mobility prediction," Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on , vol., no., pp.514-521, 20-23 Oct. 2009

[15] Ming Yu; Malvankar, A.; Foo, S.Y.; , "An Energy-Efficient Path Availability Routing Algorithm For Mobile Ad Hoc Sensor Networks," Communications, 2006. ICC '06. IEEE International Conference on , vol.4, no., pp.1885-1890, June 2006

[16] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, “A high-throughput path metric for multi-hop wireless routing,” in Proc. ACM MobiCom 2003, , San Diego, CA, Sept. 2003

[17] Y. Ledvitch and A. Segall,, Threshold Related Throughput – a new criterion for evaluation of Sensor Network Performance, Ad-Hoc Networks Journal, Special Issue on recent research directions in wireless ad hoc networking, Vol. 5, Issue 8, pp 1329-1348, Nov. 2007