Forking Agents in Sensor Networks

Thomas Haenselmann, Wolfgang Effelsberg

{haenselmann,effelsberg}@informatik.uni-mannheim.de.de

Abstract: Information about local events are to be distributed by agents in a sensor network. The agents are sent out traveling hop-by-hop into random directions, leaving a path of information behind them in the nodes visited. An arbitrary node requesting for information also emits an agent traveling into a random direction. Though only a fraction of all nodes will contain the requested information the high probability of a requesting agent intersecting the path of an informing colleague is exploited to find the information. A major problem of this approach called *Rumor Routing* is that it produces a highly non-uniform distribution of information. We have tackled this problem by introducing *Forking Agents*.

1 Introduction

In the context of sensor networks many algorithms for medium access, routing and data dissemination have to be unrolled once again with a particular emphasis on energy consumption and sparse resource utilization. E.g., in the context of MAC-protocols an estimate for the upper bound of energy efficiency has been suggested by El-Hoiydi [EH02]. Further improvements can only be achieved by decreasing the amount of data that has to be sent. On the other hand the value of a sensor network depends on the availability of the data measured.

This problem has been addressed with an approach called *Rumor Routing* which was proposed by Braginsky and Estrin in *Rumor Routing Algorithm for Sensor Networks* [BE02]. We will describe, simulate and analyze this approach in section 2 in order to show its benefits and shortcomings. In Section 3 we will introduce an improvement which significantly increases the efficiency of Rumor Routing and we will evaluate the gain in Section 4 based on our simulation which we also offer as download.

2 Previous Work

2.1 The basics of Rumor Routing

Rumor Routing [BE02] assumes that there are some nodes in a sensor network which sense particular events while other nodes request for this information. The requesting node will

usually neither be sure about the existence nor of the location of these particular events [XHE01]. So like it is often true for sensor networks a node is not addressed by its identity or location [NI97] but by some specific attribute which it may possess only for a limited period of time [RKS+03]. In contrast to other routing approaches like *directed diffusion* [IGE00, KK00] here every node can be a potential sink of information.

A straight forward implementation of bringing requesting and sensing network participants together would be to flood either the requests or the events [LMM99]. If the network senses a high number of events these should remain silently within the nodes while a smaller number of requests could be flooded. Vice versa, if many nodes issue requests which can only be delivered by few of them, flooding the events themselves is more advantageous as most nodes are interested in the bypassing news.

All situations in between the two extremes of either flooding events or requests are suitable candidates for Rumor Routing. An event is expected to influence a cluster of nodes. This cluster will generate one or more packets which travel the net. These packets are called agents. The purpose of an agent is to inform random nodes about a particular event along its journey until its TTL (time to live) has expired. Each time the agents enters a foreign node it leaves a trace in the node's event table. This trace could consist of a key or hash value of the actual data, how many hops it traveled so far and a reverse pointer of the node it came from. Nodes being positioned directly within the field of influence of the event have a hop counter of zero.

A requesting node sends a similar agent in an arbitrary direction of the network which only contains a call for specific information. While it blindly stumbles through the net it hopes to sooner or later cross the path of an informing event-agent. Therefore it will search each visited node's event table for the information it needs. In case of a hit it knows how to proceed because the former agent left a backward reference to the node it came from. After the first successful hit the requesting agent can track the path back to the event it is interested in.

2.2 How does an agent move?

Intuitively, it would be optimal if an agent moved away from an event as fast as possible. Figure 1 (A) shows four simulated scenarios according to the table in Figure 2 using Rumor Routing. The TTL of each agent was set to 1000 in all simulation runs. It can be seen from the first column of the table and from the corresponding upper left corner of Figure 1 (A) that the agent moved to another cell 1000 times while only a set of 436 different cells were visited in total. The fact that the average cell was visited more that twice is explained by the fact that the agent did not perform any consideration about what cell to visit next. The authors of the original work also encountered the problem of a small expansion due to multiple visits.

The solution to this problem is their so-called *straightening algorithm* which stores a set of recently seen nodes and travels to new ones in the first place. The name of the algorithm is not suggested by the simulation runs immediately. However, the reader should keep in

mind that finding a straight line away from an event is no trivial task if each node has no better topology information than an abstract neighborhood table.

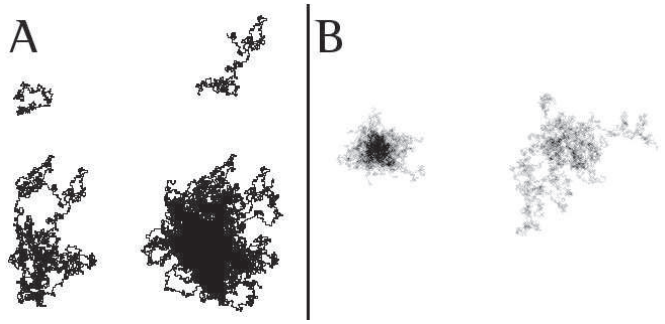

Figure 1: (A): *Rumor Routing* was simulated with varying parameters like the number of agents or their behavior according to the table in Figure 2. Increasing the number of agents spreads information further into the network but always leaves the biggest cluster of informed nodes around the origin of the event. (B): Again, Rumor Routing was simulated on the left using 16k packets to distribute information and the extension *Forking Agents* is shown on the right using only 12k packets according to the table in Figure 3. Forking Agents was able to spread the information more equally in the network with less packets. In this simulation darker shades reflect the number of (unnecessary) visits of a node.

no. agents	1	1	5	30
straightening	no	yes	yes	yes
TTL (hops)	1000	1000	1000	1000
transmissions	1000	1000	5000	30000
unique visits	436	964	4127	9884
Figure 1 (A)	up/le	up/ri	lo/le	lo/ri

Figure 2: In four simulation runs the number of agents was varied as well as the agent's awareness on preferably visiting unique neighborhood nodes in the first place (=straightening algorithm). Increasing the number of agents to five yields a coverage of a larger area. Starting as many as 30 agents however causes an mount of network traffic which is no more justified by the gain on covered area.

3 Forking agents

The efficiency of Rumor Routing lies in the fact that knowledge about an event can be distributed relatively fast within a network. This is especially true if the number of agents and their TTL ist tuned for the size of the network and the size of the cluster that perceived an event. If the number of agents is set too high simply flooding the network can become more efficient compared to agents that visit a cell multiple times.

This is were our improvement of *Forking Agents* comes in. It solves in particular the task of distributing agents more equally. Figure 1 (B) shows the problem of Rumor Routing

(on the left side) in case of a larger number of agents. In the simulation a cell was darkened each time an agent passed by. We assumed that all agents were issued by an event that is located near the middle of its cluster. As a consequence the inner cells are visited very often. Rather than clustering in the middle it would be more beneficial if the high numbers of multiple visits could be shifted into regions of the network further away from the event. The ultimate aim of the routing approach is to popularize an event all over the network.

The idea of *Forking Agents* is to let an event emit a number of "master agents". Master agents are transmitted safely from hop-to-hop which leads to two packets, one for the information and one for an acknowledgment (ACK). On their way though the network each of them forks a predefined number of times. The forking process results in a child agent which acts like any other agent but which will not produce children itself. Children are not protected by ACKs so they can get lost. The benefit of Forking Agents is that they do not have to be emitted from within the same initial place and that all saved ACKs can be replaced by more valuable event-packets. Figure 1 (B) shows the plot of a simulation. On the left side Rumor Routing was used with 64 agents, on the right Forking Agents were used with four master agents (see the table in Figure 3 for the simulation details). Obviously the extension yields a smoother distribution of agents due to the fact that not all of them have to start from the same place.

Dark shades of a cells reflect a high number of multiple and unnecessary visits, so lighter cells are more advantageous. As the agents advance into a further distance they keep leaving bigger holes in the network with nodes which have never been visited. This does however not decrease the efficiency of the overall approach a lot. A querying node does not necessarily have to reach the trail of an agent in one hop only. Leaving bigger gaps between informed nodes and imposing a minimal burden on a requesting agent is more efficient with regard to the overall performance and memory usage within the nodes.

Unfortunately, there is no guarantee that a master agents will carry its child agents far away from the event. In the worst cast it will circle around its homeland event an emit all its agents there. However, there is a chance that the master-agent will travel further away. Though this chance is decreasing with increasing distance some child agents will profit from that. Another improvement can be made if we allow child agents only after a minimal journey. Even if a master-agent travels away from an event in a straight line it will take n hops to gain a distance of n hops from the event in the optimal (and unlikely) case. In order to carry the child agents away from their homeland event it makes no sense to release them sooner than those n hops. In the simulation good results were obtains for instances of n greater than 1/4 of a node's TTL.

4 Evaluation

It is important to emphasize that Forking Agents do not spread news further away under all circumstances. Rumor Routing can itself achieve the maximum distribution of information in the special case of using a single agent only. The downside of a single agent is that it will easily get lost in a radio network. While receiver ACKs could prevent this the increase

of network traffic does not carry substantial information. In our simulations increasing the number of (child-) agents and taking the risk of losing some was far more efficient. So the benefit of Forking Agents over Rumor Routing is that it allows a larger number of agents while still distributing them more equally over the network.

$$E_P = 2 \sum_{k=1}^{\infty} kP^{k-1}(1 \quad P) \tag{1}$$

As mentioned above only master agents are transmitted safely. In order to account for that we multiply each master transmission with the expected number E_P of packets needed including all retransmissions (see Expression 1). The factor 2 accounts for the packet itself and the ACK and the sum accounts for all subsequent failures. The probability P of either a lost packet or an ACK was set to 5% in the simulations. When choosing larger loss rates the number of agents used for Rumor Routing had to be increase significantly in order to keep some alive till the end of the simulation. The increase of traffic caused by securing the master agents was by far compensated by the better distribution of agents resp. by the decrease of multiple visits within the same cell. Table 3 shows the settings of the simulation of Rumor Routing and our approach.

approach	Rumor Routing	Forking Agents
no. agents	64	16
rect. area	6776	20732
no. packets	16422	12802
avg. visits	5.92	2.98

Figure 3: Simluation parameters for the evaluation of *Rumor Routing* and *Forking Agents*

The number of agents was set to 64 in Rumor Routing and to 4 master agents in Forking Agents. Each of them was allowed to fork four times. We decreased the number of agents in our approach to 16 in order to obtain approximately the same number of packets transmitted. The number of packets needed by both algorithms (16k and 12k) had to be more or less equal to be able to compare the results.

Considering only those nodes being visited at least once the average node was visited about 3 times with our extension and about 6 times using the classical approach. At the same time the area visited was increased from 6776 to about 20000 cells.

The software used for our evaluation can be obtained at
http://www.informatik.uni-mannheim.de/~haensel/forking_agents.tgz

5 Conclusion and Outlook

In our approach nodes directly sensing an event in a sensor network send out loss-protected master agents to make the event popular. After these master agents have gained some distance from the event they start to distribute child agents which have no loss-protection.

The smaller number of agents starting from within the same event and place helps to distribute a larger number of child agents into vast areas of the network. The chance of multiple visits can be decreased significantly. Generally there is no need to limit the approach to master- and child agents. We would like to generalize it for using an arbitrary number of levels of child agents into an optimal tree-like structure.

References

[BE02] D. Braginsky and D. Estrin. Rumor Routing Algorithm for Sensor Networks. In *Proceedings of the First Workshop on Sensor Networks and Applications*, pages 1–12, Atlanta (GA), USA, October 2002.

[EH02] A. El-Hoiydi. Aloha with Preamble Sampling for Sporadic Traffic in Ad Hoc Wireless Sensor Network. In *Proc. IEEE Int. Conf. on Communications*, pages 3418–3423, New York, USA, April 2002.

[IGE00] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In *Proceedings of MobiCOM*, Boston (MA), USA, August 2000.

[KK00] B. Karp and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. *Proc. of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000)*, 8:4:243–254, August 2000.

[LMM99] M. Lin, K. Marzullo, and S. Masini. Gossip versus deterministic fbooding: low-message overhead and high-reliability for broadcasting on small networks. Technical Report CS1999-0637, University of California at San Diego La Jolla, CA, USA, 1999.

[NI97] J. C. Navas and T. Imielinski. GeoCast: Geographic Addressing and Routing. *Proc. of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 1997)*, 8:4:66–76, August 1997.

[RKS+03] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and Yu F. Data-Centric Storage in Sensornets with GHT, A Geographic Hash Table. *Mobile Networks and Applications (MONET), Special Issue on Wireless Sensor Networks*, 8:4:427–442, August 2003.

[XHE01] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed Energy Conservation for Ad-hoc Routing. *In Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (ACM MobiCom)*, July 2001.