Quorum-based geographically static data storage in ad-hoc networks

G.H.Owen and M.Adda

School of Computing, University of Portsmouth,
Buckingham Building, Lion Terrace, Portsmouth, PO1 3HE
E-mail: {gareth.owen, mo.adda}@port.ac.uk

Abstract

Large-scale wireless ad-hoc networks are formed automatically and allow nodes to communicate over greater distances. Every node acts as a router forming a complex network, which is simplified by the use of geodesic routing. However, a method to discover a node's location is needed for geodesic routing. In this paper we propose two algorithms which could support location servers in ad-hoc networks and examine their performance.

Keywords

Ad-hoc networks, atomic memory, geographically static data storage, quorum systems, location servers.

1. Introduction

Ad-hoc networks are formed automatically without any pre-existing infrastructure. These networks may consist of various types of mobile devices such as PDAs, laptops, cell phones and other mobile computing devices. Each node in a wireless ad-hoc network performs everyday computational tasks as well as communication routing. Due to the lack of dedicated routing devices, the nodes must route traffic for each other to enable connectivity.

We believe that large-scale ad-hoc networks can be realised, given multi-radio nodes (Adda et al. 2005) and locality of traffic to overcome the problems of diminishing throughput (Gupta & Kumar, 2000). Users will tend to communicate with those in their local area most of the time and very little traffic will be national or international, which will further enable scalability (Li et al. 2001). It is possible to foresee a national or international ad-hoc network replacing cellular networks and the internet; however, several obstacles need to be overcome before this can be realised, one of which this paper attempts to address.

Large-scale ad-hoc networks need a method of finding the nodes that differs from small ad-hoc networks. This is because the standard methods do not scale much above a few hundred nodes, due to the limitations in the discovery mechanisms. Routing in large-scale ad-hoc networks is achieved by using geodesic (Hubaux et al. 2001; Ko & Vaidya, 1998) routing so that no initial path discovery is necessary. Several approaches have been proposed using nodes within the network to act as location servers serving up the whereabouts of each other. Stojmenovic (1999) proposed a method where a node broadcasts its location vertically and horizontally but not diagonally. This enables any node to query the location by sending packets in a vertical or horizontal direction until it encounters a location server. However,

this is not practical on a global scale as the information would be broadcast many thousands of miles. Li et al (2000) proposed a complicated approach based upon grids where a certain number of nodes within an increasing size of grid contained the location information. This system is overly complex and difficult to query in a large network. Finally, Giordano and Hamdi (1999) proposed the home region approach, which designates nodes within a given radius of a particular point to act as the location servers. To query this information the source node simply sends a packet to that location. They did not address issues of node mobility and how to avoid data being removed from the home region as nodes move.

This research contributes to the field of location servers in ad hoc networks. The technique for addressing nodes within a network is introduced followed by two techniques for quorum based location servers. The main difference from the previous works is that this proposed scheme deals effectively with node mobility by expanding on the non-quorum approach proposed for dealing with mobile location servers (Owen and Adda, 2005). The performance of the techniques is then analysed and compared by looking at survivability, network, memory and processing overhead. The location servers must always be accessible and the overhead incurred must be low enough so that every node may have a group of location servers without degrading the performance of the network.

Section 2 will describe two variants of the algorithm for providing quorum based geographically static data storage. Location servers could utilize this technique in large-scale ad-hoc networks. Section 3 will detail our results and analysis of the simulation of the two algorithms. Finally, sections 4 and 5 will detail future work and the conclusions of this paper.

2. Geographically static data storage (GSDS)

In this section, we develop two algorithms that provide a quorum based approach to geographically static data storage. The two algorithms are similar but differ in the method they use to maintain the quorum. We develop a unicast and broadcast variant with the emphasis on determining how the two perform against each other.

This algorithm is an adaptation of the work undertaken that showed that geographically static data storage was possible (Owen and Adda, 2005). However, no implementation of a quorum was undertaken and the data was particularly prone to losses through nodes' failures. Our algorithm, described in this paper, provides a type of quorum system designed for storing data at a geographically static point in an ad-hoc network. That is to say, should one wish to store an item of data in the office, nodes that are in, or around the office should always keep a copy of the data item. Data is queried by sending a packet to the geographic point in the hope that a node along the path to this point or within the vicinity of the point will hold a copy of the data. If a data item is not where it should be, or not near the point then the data is effectively unavailable at that period.

The first hurdle is that a node needs to know the location of its neighbours so that it can perform geodesic routing. Füßler et al. (2003) has implemented a beaconless scheme though this introduces unnecessary delays into the network, especially when the cost of beaconing in modern wireless networks is negligible. In this paper, we describe the development of our algorithms that require a beacon based mechanism that allows every node in the network to broadcast its ID and location information every five seconds. To support the quorum system, we embed details of the data items held at a node into every beacon frame. This will allow

the algorithms to make decisions about the duplication or migration of data items, without querying nodes on-demand.

Each quorum needs a master to co-ordinate the replication of data. Every node is responsible for deciding whether any of the data items it has are eligible for promotion or demotion. If a node can identify that it is not the closest to the geographic point, then the node will check its received beacon data to see if the closer node already has a data item. In this case, the node demotes the data item to a slave; otherwise, the item is migrated to the closer node and will possibly be promoted to the master.

2.1. Unicast variant (GSDS-U)

The unicast algorithm builds on the base described above to implement replication of the data. The algorithm attempts to maintain an arbitrary number of slaves among its neighbours by unicasting a copy of the data item to them. This is achieved by monitoring the beacon data received from other nodes to determine how many slave data items there are and identify when data items are lost (e.g. due to power failures).

The algorithm is broken into three conditions, the decision to migrate, to duplicate, and to prune. The migrate condition determines whether a data item could get closer to the destination. A migration is made if the closer node does not already have a copy of the data item. It should also be noted that slaves are also migrated to prevent the need to be duplicated should the nodes leave the area. Figure 1 illustrates this as a node A enters the area and becomes closer to the master M than the node that is leaving the area, B. In this diagram, nodes hosting slave data items are labelled by S.

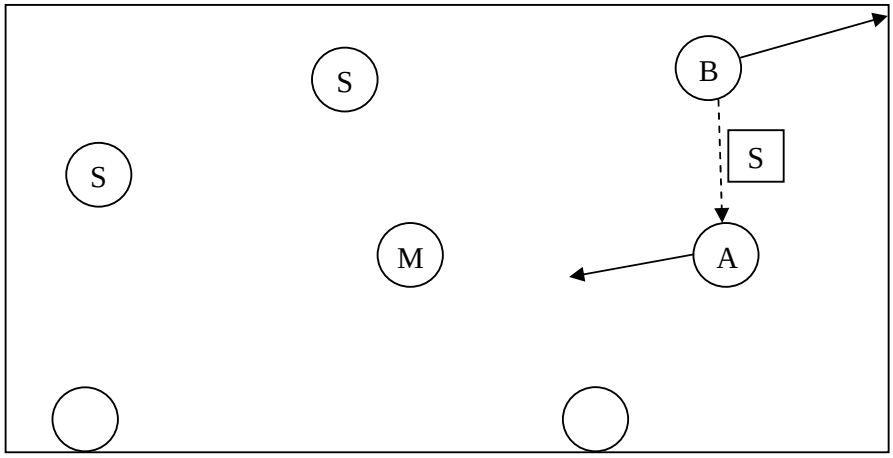

Figure 1: Illustration of slave migration

The duplicate condition determines whether the required number of slaves has been reached. This is achieved by looking at the beacon data of the surrounding neighbours. If the number of slaves is below a threshold, the algorithm unicasts a copy of the data item to another neighbour node without a copy.

Limiting the number of slaves is important to the efficiency of the algorithm as more duplicated slaves could flood the network. Therefore, each node periodically prunes its slaves by removing one if there are enough slaves within its vicinity. To avoid a situation where all the nodes prune at the same time, the algorithm staggers these by introducing a random jitter.

Given the number of migrations for all copies of the data, φ, the number of duplication packets from the master, σ, and the number of packets sent, n, we express the overhead in packets, p, in the unicast algorithm as:

$$p = \sum_{i=1}^{n} \varphi_i + \sum_{i=1}^{n} \sigma_i \quad (1)$$

2.2. Broadcast variant (GSDS-B)

The broadcast algorithm attempts to reduce the overhead introduced by the replication of the data items. One broadcast packet is equivalent to many unicast packets at the duplication phase. This remark is the foundation of the broadcast algorithm.

The algorithm attempts to maintain its quorum by the master periodically broadcasting a copy to all the neighbours. All neighbours that overhear this store a copy of the data item and assume the role of a slave. Slave data items are mobile in this algorithm, but because more nodes host them they may induce a higher migration overhead.

The broadcast algorithm is also broken down into the same three conditions. The first condition works the same as the unicast algorithm by migrating data to a closer node. The second condition is also the same with the exception that the node broadcasts a copy to all of its neighbours, as opposed to unicasting a number of times. This then relies on the third condition to reduce the number of slaves to a reasonable level. This is performed in an identical manner to the unicast algorithm with the pruning of data items staggered to avoid a mass-loss of slaves.

2.3. Simulation scenario

The simulation consists of a network of 200 nodes using the GloMoSim simulator, which was chosen due to its good scalability. GloMoSim's parameters are set to provide a free-space propagation model using a standard radio and the 802.11b MAC protocol. The Random Waypoint Model is used to simulate walking and cycling individuals in an area of 1km².

The simulation simulates node power failures by deleting the entire node's data and leaving the node unavailable for use for a random period. At five-second intervals, each node generates a random number, and if this is larger than a chosen figure then that node simulates a power failure. The total number of node power failures at any given time is denoted by β(t). Given the interval in seconds, λ, the probability of power failure per node, P_i, and the number of nodes, n, we can express the total power failures as:

$$\beta(t) = \frac{t}{\lambda} \sum_{i=1}^{n} P_i \quad (2)$$

The model of the free space loss is used which may not provide an accurate representation of a city centre or any built up area. It is expected that the node degree would decrease in a real experiment and would therefore decrease survivability. However, the study simulates a variety of node power failure scenarios and it is expected that real life scenarios would have a low node power failure rate in comparison to the top end of our simulations. Real world

applications would comprise of static nodes such as wireless access points and desktop PCs that could participate in the process; given an intelligent duplication algorithm, survivability should increase by choosing candidates that are more appropriate.

3. Results

Several simulations were performed to analyse the survivability of the algorithms under adverse conditions where nodes are experiencing random power failures. Each simulation was performed 10 times to validate the results, and the average of these 10 simulations represents one point on the graph. Looking at the results in Figure 2, we can see that even when every node in the network powers off at least once a minute, up to 20% of the data items are still able to survive when using the unicast algorithm. However, the broadcast algorithm performs better than the unicast algorithm because while the pruning process is working, the algorithm has many more duplicates.

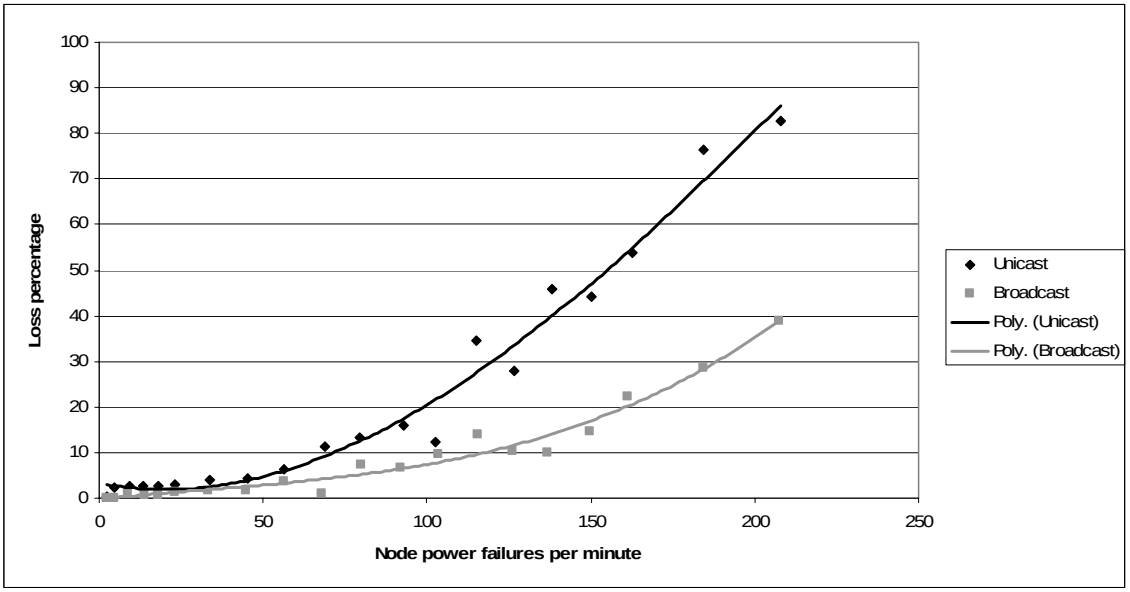

Figure 2: Survivability of the two algorithms

Memory and processing overheads per node measure on average the number of data items held on each node at any given time. The memory and processing overheads scale with the number and size of the data items. As one would expect, the memory overhead with the broadcast algorithm is higher because it is duplicating to more nodes than the unicast algorithm (Figure 3).

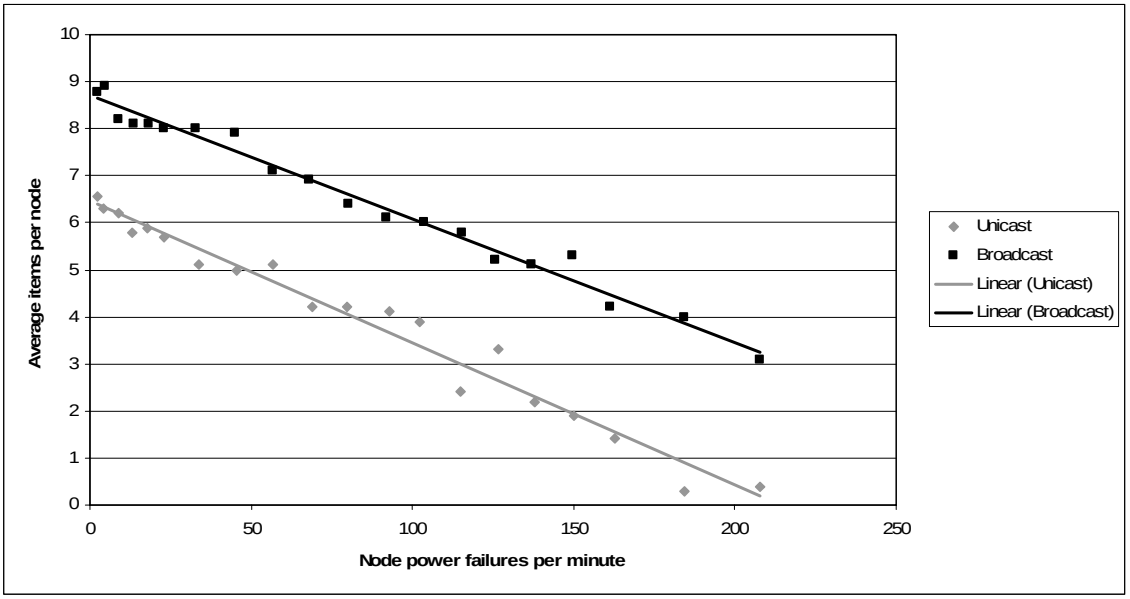

Figure 3: Memory/processing overhead

Unexpectedly, Figure 4 shows that the algorithms perform very similarly when it comes to network overhead. We expected that the broadcast algorithm, while saving on the initial duplication cost, would incur a higher overhead because of the migration of many more data items. However, because there are more duplicates fewer places are unoccupied for them to migrate to and as such, the duplicates tend not to migrate during the period in which the pruning takes place.

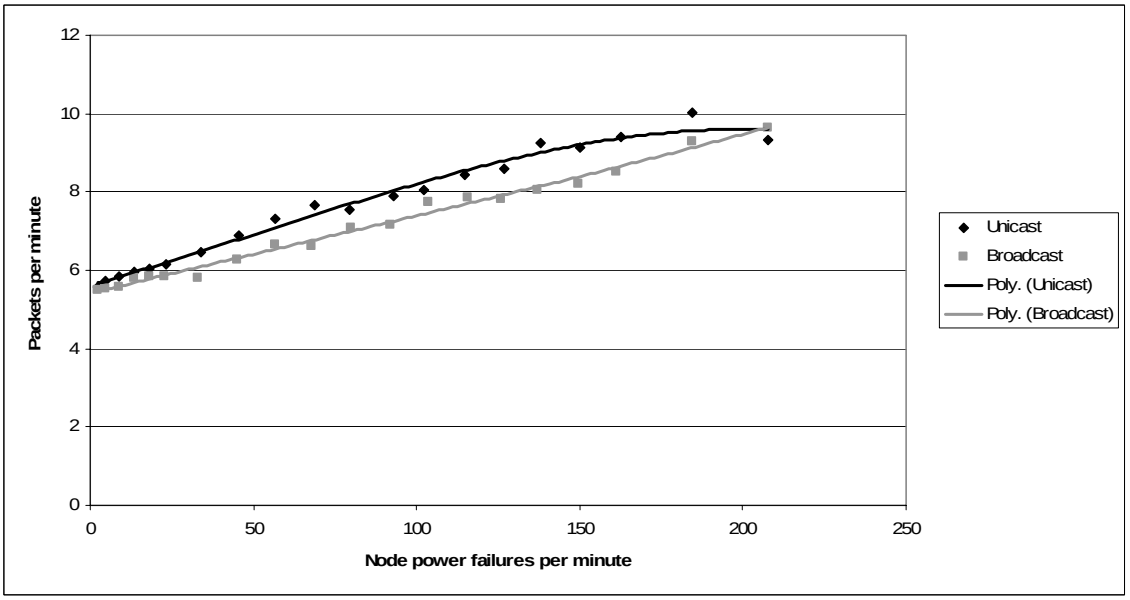

Figure 4: Network overhead

We also simulated the above characteristics with different percentages of mobile nodes, such as networks with static access points and similar devices; however, this had no effect on survivability. The network overhead declined as expected where there were less mobile nodes, and in the case of a static network with no power failures, the network overhead was

zero after the initial set-up cost. This demonstrates that the algorithm is able to adapt to the existing network conditions imposing the minimum overhead for the specific scenario.

4. Future work

This work gives an understanding of how quorum systems perform under certain ad-hoc networking scenarios. Further work could improve on this by adapting the broadcast algorithm to broadcast to only a small number of nodes. Nodes could decide probabilistically whether to keep them depending on the node degree, which may decrease the overhead incurred through migrations. Both approaches have advantages and disadvantages and perhaps a future approach could combine the two into a hybrid algorithm that may be able to maintain high survivability while keeping the overhead low.

Future work will also look at how the power consumption of the protocols can affect their availability. It is expected that the protocol with higher power consumption would more strongly affect survivability. This may change the outcome of the survivability experiments. In addition, one needs to look at using these approaches as location servers for supporting applications over ad-hoc networks.

This technique of achieving geographically static quorums in ad-hoc networks could have more wide reaching applications such as location-based services and wiki-style environments.

5. Conclusions

This work has shown that quorum systems in ad-hoc networks are possible and do not have to incur a large overhead. We proposed two techniques to achieving this and both approaches have their benefits and pitfalls. We showed that the broadcast algorithm survived better but that it also incurred a bigger memory/processing overhead. However, the extra overhead incurred was very small. Therefore, we conclude that the use of the broadcast algorithm in terms of survivability is the preferred choice as it performs significantly better without an equally significant overhead.

The emphasis of this work has been on location servers, but the algorithms developed could have many more wide reaching applications in a large-scale ad-hoc network. It is expected that large-scale ad-hoc networks will become commonplace and quorum based systems will be essential in providing reliable distributed services.

6. References

Adda, M., Owen, G., Kasassbeh, M., Paraskelidis, A., Peart, A. (2005). Communication Issues in Large Scale Wireless Ad Hoc Networks. In *proceedings of the Athens Institute for Education and Research conference* (pp 299-313). ISBN 960-88672-3-1.

Füßler, H., Widmer, J., Käsemann, M., Mauve, M., and Hartenstein, H., (2003). Beaconless position-based routing for mobile ad-hoc networks. *Technical Report TR-03-001*, Department of Computer Science, University of Mannheim

Giordano, S., Hamdi, M., (1999). Mobility Management: The Virtual Home Region. *Technical Report No. SSC/1999/037*, EPFL.

Gupta, P., Kumar, P. R., (2000). The Capacity of Wireless Networks. *Journal of IEEE Transactions on Information Theory*, 46(2), 388-404..

Hubaux, J., Gross, T., Le Boudec, J., Vetterli, M., (2001). Toward Self-Organized Mobile Ad Hoc Networks: The Terminodes Project. *Journal of IEEE Communications*, 39(1).

Ko, Y.B., and Vaidya, N.H., (1998). Location-aided routing (LAR) in mobile ad hoc networks. *In proceedings of the ACM/IEEE 4th Annual International Conference on Mobile Computing and Networking*.

Li, J., Blake, C., De Couto, D., Lee, D., Morris, R. (2001). Capacity of ad hoc wireless networks. In *proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking* (pp 61-69).

Li, J, Jannotti, J., De Couto. D., Karger, D., and Morris, R., (2000). A scalable location service for geographic ad hoc routing. In *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking*, (pp 120-130).

Owen, G., Adda, M. (2005). Feasibility of geographically static data storage in ad hoc networks. In *proceedings of the IADAT 2nd International Conference on Telecommunications and Computer Networks*, (pp 172-176).

Stojmenovic, I., (1999). A scalable quorum based location update scheme for routing in ad hoc wireless *networks*. *Technical Report TR-99-09*, University of Ottawa.