An Efficient Resilience Mechanism for Data Centric Storage in Mobile Ad Hoc Networks

Dominique Dudkowski, Pedro José Marrón, and Kurt Rothermel
University of Stuttgart
Institute of Parallel and Distributed Systems (IPVS)
70689 Stuttgart, Germany
{dudkowski|marron|rothermel}@informatik.uni-stuttgart.de

Abstract

Data Centric Storage (DCS) is a powerful storage paradigm for wireless ad hoc networks. In mobile ad hoc networks (MANETs), however, the mobility and varying density of nodes may significantly impact the efficiency of data access and the level of data consistency for existing DCS mechanisms. In this paper, we propose an efficient resilience mechanism for data centric storage that supports DCS in mobile environments. We introduce a novel indirection strategy that enables us to distinguish the storage of data at dedicated server nodes from the storage of additional information to locate these servers. Our approach places server location information dynamically in strategic parts of the network based on its current topology. Combining our server location advertisement with any geographic routing protocol, we provide a robust data update and query processing technique for data centric storage in MANETs. We show analytically and by means of experimental evaluations that, despite the additional indirection during packet forwarding, our approach provides superior storage and retrieval performance than the original DCS algorithm even for large amounts of dynamic data.

1 Introduction

Data centric storage (DCS) is a powerful concept for data management in large-scale wireless ad hoc networks. Originally proposed by Ratnasamy et al. in [13] in the area of wireless sensor networks, it provides efficient and scalable mechanisms for storage and retrieval of data. The primitives of DCS may be used to implement a variety of more complex architectures, such as the distributed index for multidimensional range queries discussed in [11], making it a fundamental component for data management in wireless ad hoc networks.

The key concept of DCS lies in the use of a geographic hash function that computes a hash value for each data type encoding the location of data stored in the network. Assuming that the hash function is well known, storage and retrieval operations on the same data type will always be mapped to the same location and, by extension, to the same network node (called *home node*).

In order to make DCS work properly, the local routing strategy must guarantee that requests are consistently routed to the network node responsible for the storage of the relevant subset of data. Two types of strategies have been proposed so far in the literature: The first one tries to determine the *home node* of the reference location based on geometric distance. The DCS approach in [13] uses such a strategy. The second type uses geometric cells inside of which nodes are recruited for data storage. The rendezvous regions approach proposed in [14] belongs to this group.

In MANETs, however, these types of strategies cannot cope well with the mobility of network nodes. The first type of approaches assumes the existence of a perimeter around the reference position (see [7, 13]), and identifies a home node by making a full turn around it. This approach fails if a perimeter is too long, which may occur in the case of networks with low node density or in the event of network partitions. The main issue of the second type of approaches has to do with the overhead associated with the management of cells that do not contain any network nodes. In both types of approaches, data consistency is immediately affected by the underlying characteristics of MANETs, requiring additional management at increased communication costs.

To overcome the problems related to perimeters and cell-based structures, we propose a novel resilience scheme designed to support the storage of large quantities of frequently accessed data in mobile ad hoc networks according to the DCS paradigm. The key concept of our approach is a novel indirection strategy based on the separation of stored data from server location information required to determine which servers store a particular data item. Since data itself is stored on dedicated server nodes, it is possible to build replication algorithms easily on top of our resilience scheme in a similar way to SR-DCS [13] or R-DCS [5]. While at first sight, an extra indirection step might suggest a larger communication overhead, we will show that for large amounts of data subject to frequent requests, the indirect scheme outperforms the previous DCS scheme proposed in [13] in terms of communication costs.

The remainder of our paper is structured as follows. In Section 2 we present previous work related to data centric storage. In Section 3 we describe the system model we use to discuss the details of our resilience scheme for DCS in Section 4. We provide suitable analytical considerations and simulation results of our mechanism in comparison to DCS over GPSR as proposed by Ratnasamy et al. in Section 5 and conclude our paper in Section 6.

2 Related Work

Let us now discuss previous approaches in the context of data centric storage with respect to their applicability in mobile ad hoc networks. We concentrate on the ability of each approach to cope with node mobility and varying node densities, large quantities of data requested frequently, and data consistency.

Data Centric Storage (DCS) was originally proposed for wireless sensor networks by Ratnasamy et al. in [13, 15]. Their approach uses the Greedy Perimeter Stateless Routing protocol (GPSR) to locate a home node responsible for storing data [7]. The authors extended DCS by Structured Replication (SR-DCS) to load-balance data associated with the same key to multiple home nodes at different locations. To achieve data consistency in the presence of limited node mobility and node failures, DCS employs the Perimeter Refresh Protocol (PRP) to replicate data around a perimeter in addition to the home node. In [5] Ghose et al. propose an extension to DCS which they call Resilient Data Centric Storage (R-DCS). Using replica and monitoring nodes, they optimize knowledge about data availability in different regions and the exchange of data between different cells to provide a certain degree of replication. Similar to DCS, they apply GPSR to locate storage nodes and they optionally support PRP to recover data at least partly from departing and failing nodes. Especially in MANETs, both approaches involve increased communication costs for each request routed by GPSR and each refreshing cycle of PRP.

Various improvements of geometric face routing proposed in the literature follow the idea of GPSR and allow the localization of a home node more efficiently. Chen et al. propose On-Demand GPSR (OD-GPSR) in [2] to optimize perimeter traversal. However, their approach works well only for stationary networks, because they store control information that can be used to detect previously traversed perimeters. Other face routing protocols, such as AFR and its extensions by Kuhn et al. (e.g. in [10, 9], and the Face-Aware Routing (FAR) in [6], can also be found in the literature. Complementary, planarization techniques based on the Gabriel Graph and the Relative Neighborhood Graph were proposed in [7] for GPSR, and the Cross-Link Detection Protocol (CLDP) proposed in [8] to optimize face routing for both unit and non-unit disc graphs. Although these concepts provide significant improvements to face routing with respect to packet delivery success ratio, they do not provide a solution to determine a practical home node in the presence of open and long perimeters in mobile networks.

As opposed to locating a node closest to the reference location to play the role of a home node, many authors have introduced modifications to DCS that use nodes within fixed cells to store data. In [16] Tamishetty et al. eliminate the need of perimeter routing by relying on fixed-sized geometric cells inside of which the information of available storage nodes is disseminated by a flooding mechanism and stored on a number of nodes. A similar idea is used by our implementation of probabilistic spatial queries in [3]. Seada and Helmy introduce Rendezvous Regions (RR) in [14] that relies on a fixed subdivision of the total area into cells as well. Araujo et al. propose Cell Hash Routing (CHR) in [1] that resembles the aforementioned approaches, also making the assumption that cells have a globally known and predefined shape. Although these approaches eliminate the need of perimeter routing and thus, the localization of a particular home node, their dependance on fixed areas leads to the problem of insufficient node population that impacts their proper operation.

In contrast to the discussed approaches, we provide a robust and *practical* mechanism to support DCS in mobile ad hoc networks during the recruitment and localization of suitable storage nodes. Our strategy does not require cell-like structures of the service area, and it does not rely on full perimeter traversals. Thus, we are able eliminate both the cases of non-populated cells and the problems related to full perimeter traversals, as described in more detail in the following sections.

3 System Model

Let $A \in \mathbb{R}^2$ be a service area containing mobile nodes v_i, which are able to communicate at a distance given by the transmission range r_{tx}. Each node is located at a physical position $\mathbf{x}_i(t) \in A$ which is subject to change over time. The position may be obtained algorithmically (e.g. using [12]) or from positioning systems, such as GPS [4]. Nodes can take the following roles: *relay*, *server*, *coordinator* or *client*. By definition, all nodes act as relays, in order to forward packets using geometric routing. A subset of nodes may function as servers, which are responsible for storing data. Coordinators store information about individual servers that may be consulted during routing to locate a particular server. Finally, clients are nodes that issue requests, either data updates or queries.

Regarding the data model, we assume a logical data space $\mathbb{D}$ with data objects $o_j \in \mathbb{D}$. Each object has some dynamic state associated with it (e.g. a temperature reading). We define a set C of fixed *coordination points* $\mathbf{c}_i \in C$, which may play the same role as locations in DCS [13], represent a reference position (e.g. the center of a cell) to support rendezvous regions [14], or our architecture for spatial queries in [3].

We assume that subsets of $\mathbb{D}$, which may be non-disjoint, are associated with a coordination point by a relation $R \subseteq \mathbb{D} \times C$. The relation R may be static, for instance implementing geographic hash functions like in [13, 14], or dynamic, as in [3]. For ease of discussion, we assume a static relation R that associates each data object $o_j \in \mathbb{D}$ with a single coordination point. Furthermore, we assume a relation between servers and coordination points. If a server is associated with coordination point $\mathbf{c}_i$, it is responsible for storing the subset of data related to $\mathbf{c}_i$ by R. In the following, we assume that one server is associated with each coordination point. More complex relations, e.g., the association of several servers with a single coordination point, are possible and follow similar ideas found in structured replication [13] or R-DCS [5].

4 Resilience Mechanism for DCS

The mechanism that we now present in detail is designed to provide resilience to node mobility for data centric storage in MANETs. The general idea is to adapt the local routing strategy of existing face routing protocols to provide a practical and effective solution for home node localization. We call this strategy Bidirectional Perimeter Routing (BPR) and discuss it in detail in Section 4.1. In Section 4.2 we introduce our concept of *server advertisement*. It uses BPR to distribute the information required to locate a server during packet routing. We then propose our indirection strategy during the forwarding of client requests in Section 4.3 and show how we achieve efficient and robust packet routing from clients to servers.

Note that, for mobile servers, we also require a mechanism to migrate data from one server to another node once a server moves away from the coordination point it is related to. We omit the details of our migration protocol and refer to [3] for additional details. However, note that the complexity analysis and experimental evaluation of Section 5 take into account the cost of server migration.

4.1 Bidirectional Perimeter Routing

Bidirectional Perimeter Routing (BPR) is a customization of geometric routing protocols that apply face routing while relaying packets towards their destination. The first geometric protocol that applied this strategy was GPSR [7]. While BPR may be used with any geometric routing protocol that uses face routing, we will use GPSR as an example in the following exposition.

Of particular interest to DCS is the operation of GPSR in an *indirect mode* to allow locating the *home node* on a perimeter around a geometric position without specifying the address of that node directly. This modification is proposed in [13] and yields the nearest node to the position in question. However, the proposed method is likely to fail in more realistic MANET scenarios (e.g., urban scenario with obstacles) where the length of perimeters may increase critically. Figure 1(a) and 1(b) show two situations in a network, in which the length of perimeters becomes critical. In Figure 1(a), position $\mathbf{c}$ lies inside of a region (sometimes called *void*) that does not contain any network nodes. In this case, a packet is routed on a perimeter around the empty region. In Figure 1(b), $\mathbf{c}$ lies outside the network partition inside of which the packet originated and, therefore, the packet travels on the perimeter that spans the inside of the partition. If the total perimeter length is greater than the residual time to live of a packet after starting at v_p, the packet will be dropped before it is able to reach v_p for the second time, which is necessary to finally decide on a home node.

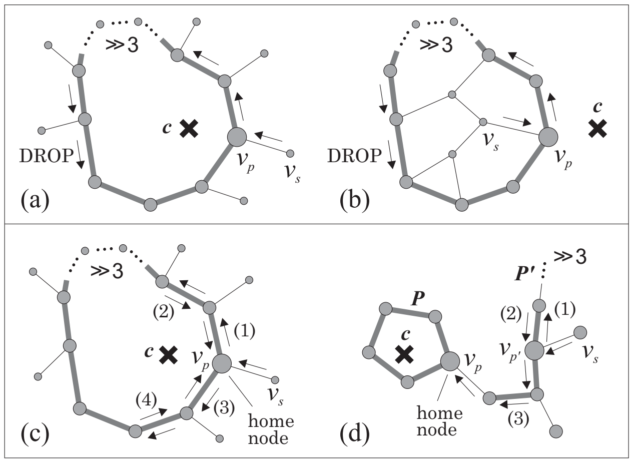

Figure 1. Bidirectional Perimeter Routing

In order to avoid having to route packets around the complete perimeter, BPR determines a *practical* home node based on the traversal of only a *partial perimeter*. For that purpose, BPR specifies a *perimeter radius* R_p, which defines the number of hops a perimeter may be traversed in one direction starting from the perimeter entry node. Thus, a *partial perimeter* is defined as the set of nodes reachable from the perimeter entry node v_p in at most R_p hops.

As shown in Figure 1(c), the modified face routing strategy of geometric routing protocols is adapted in the following way. Consider a source node v_s and a coordination point **c**. A packet begins its life in greedy mode, similar to GPSR, and is forwarded until it is not possible to find consecutive nodes closer to **c**. In this case, BPR routing switches to *bidirectional* perimeter mode at the perimeter entry node v_p, where a packet is forwarded according to the right-hand rule (1). Each hop visited on the perimeter is recorded in the packet header. Forwarding continues while each visited hop on the perimeter is further away from the coordination point than the perimeter entry node and the perimeter radius R_p is not reached. If it is not possible to find a node nearer to the destination in the first direction, the source route stored in the packet header recorded previously goes back to v_p (2) and repeats this procedure in the other direction (step (3) and (4) in Figure 1(c). Again, if we do not find a node closer to the coordination point than v_p, v_p becomes, by definition, the home node of coordination point **c**.

Our choice to use bidirectional parameter traversal is to take into account topologies that may contain highly asymmetric perimeters. Figure 1(d) shows an example of an asymmetric perimeter P'. Because it is not possible to determine a priori at the perimeter entry node which direction of a perimeter may succeed, we treat both directions equally. In Figure 1(d), even if we select a very small perimeter radius, we will be able to discover the actual perimeter P around **c**. Observe that for small and closed perimeters, such as P in Figure 1(d), BPR is able to detect these perimeters in a similar way to GPSR.

Of course we may not neglect the additional routing costs imposed by BPGR when returning on a perimeter using source routing. Later on in Section 5.1 we will justify the increased routing overhead and show how we are able to benefit from the potential of bidirectional perimeters.

4.2 Server Advertisement

Let us now discuss the server advertisement strategy. As shown in Figure 2, each server sends advertisements on a regular basis to its associated coordination point. An advertisement contains relevant information to identify currently available servers and their association with particular coordination points. That information comprises the server's ID and current position, and the ID of the coordination point the server is associated with. In the first phase, an advertisement is distributed to populate a partial perimeter using BPR in indirect mode.

In Figure 2(a), the information is sent to the nodes connected by the thicker line using bidirectional perimeter routing. The information contained in a server advertisement is

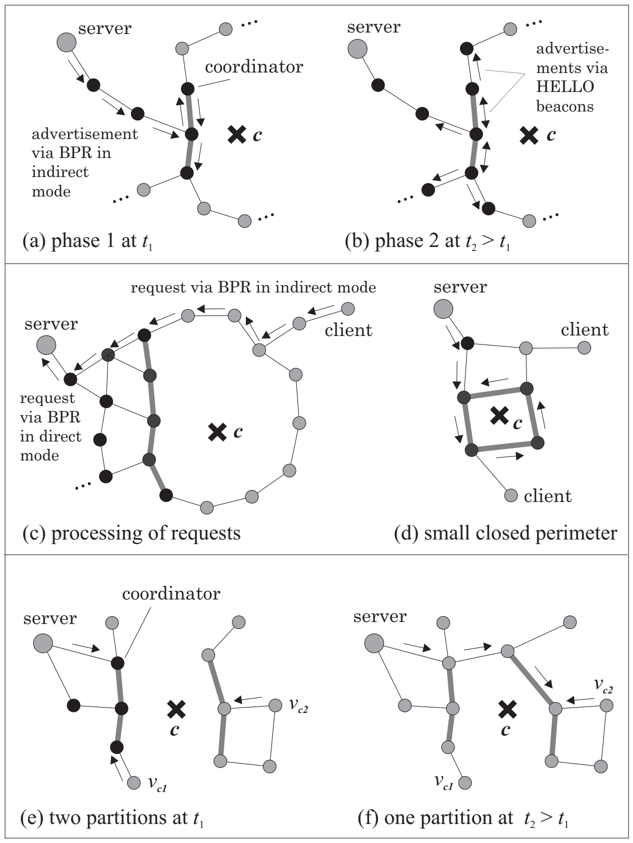
server
coordinator
advertisement via BPR in indirect mode
c
(a) phase 1 at t_1
server
advertisements via HELLO beacons
c
(b) phase 2 at $t_2 > t_1$
request via BPR in indirect mode
server
client
request via BPR in direct mode
c
(c) processing of requests
server
client
c
client
(d) small closed perimeter
coordinator
server
v_{c2}
c
v_{c1}
(e) two partitions at t_1
server
v_{c2}
c
v_{c1}
(f) one partition at $t_2 > t_1$

Figure 2. Protocol Details

stored at each visited node, together with a lifetime. Observe in Figure 2(a) that for the distribution of a server advertisement, it is not required to return to the perimeter entry node after the perimeter radius has been reached in the second direction. Observe also in Figure 2(d) that for small closed perimeters, server advertisements are distributed around the full perimeter.

While server advertisements are forwarded along perimeters, the mobility of nodes may eventually lead to a modification of perimeters. While some nodes will move onto a perimeter, coordinators will leave the perimeter. These dynamics lead to a degradation of the resilience to find server information. We propose to use existing HELLO beacons of the underlying geometric routing protocol to distribute server information to additional hops around a partial perimeter. This way, the availability of server information around the partial perimeter is increased, providing additional resilience to node mobility during client routing. Two reasons make HELLO beacons attractive. First, nodes send beacons on a regular basis for the purpose of accurate neighbor information to their neighbors in a single broadcast. This is a similar purpose to the server advertisement information which we want to be fresh as well. Second, server information is only a small piece of information, therefore, it may easily be piggybacked onto HELLO beacons. Our approach is to emit a HELLO beacon with server information immediately after a node has received a server advertisement in the first phase, to provide fresh server information to nodes in the 1-hop vicinity of the partial perimeter. To avoid unnecessary regular beacons, the next beacon is rescheduled after the beaconing interval. Using a suitable synchronization between server advertisements with regular beacons, *no additional* beacons are required. Figure 2(b) shows the nodes (coordinators C) that have received server information via a single cycle of HELLO beacons.

4.3 Processing of Requests

Client requests are forwarded to servers in two phases using any geometric routing protocol that uses face routing, e.g. GPSR or BPR for an infinite setting of the perimeter radius. We will use BPR in the following discussion. We further assume that a client includes its own location in the request. Each request is routed in phase 1 using BPR in *indirect mode* into the direction of the coordination point. At each visited node, we perform a lookup to check if fresh information is available for the server that holds the data for the requested coordination point. Consider Figure 2(c), where a client request travels around a perimeter in one direction. Eventually, it will reach nodes that hold fresh server information, which was distributed by server advertisements in the vicinity of a partial perimeter. Upon finding fresh information, the request is rerouted to the actual server via BPR in direct mode. In order to do this, the only information required are the ID and location of the server. Upon reaching the server, the request is processed and a reply is routed back from the server to the client using BPR.

To show the robustness of our resilience mechanism, we have depicted the case of a network partition in Figure 2(e). Because we always restrict the distribution of advertisements to a partial perimeter, communication costs remain fixed, even in the presence of network partitions. Clients in the same partition (v_{c1}) are able to locate that information, though clients in a different partition (v_{c2}) will not. In Figure 2(f), after some time, the two partitions join. If no additional server advertisements have occurred yet, client v_{c2} will immediately be able to locate the only information that was formerly located in the first partition. Observe that the joining of partitions implies a different partial perimeter along which consecutive server advertisements are distributed. Clients will still find the server information at the newly selected coordinators. Also, if the previously distributed server information is still considered fresh, it may be used by clients as well (e.g., by client v_{c1}).

5 Evaluation

We now present evaluation results to show the performance of data centric storage based on our resilience scheme (DCS/BPR) in comparison to DCS/GPSR as proposed in [13]. Section 5.1 presents analytical results concerning the communication costs in the tradition of [13] and [5]. In Section 5.2, we present a suitable subset of simulations to show some additional properties of our mechanism.

5.1 Analytical Study

Ratnasamy et al. have analyzed the communication costs of DCS/GPSR in [13] based on the assumption that the size of perimeters is small in comparison to the number of hops required to route to a perimeter (global analysis). Because this assumption does not apply to MANETs, the local costs around each coordination point make up a substantial part of the overall communication costs. Refer to Table 1 for a list of the symbols used in the following analysis.

n	total number of network nodes
$\bar{v}$	average speed of nodes in m/s
m	total number of data objects
r_u	number of updates per data object and second
p	capacity of data packets in number of data objects
r_a	rate of server advertisements
r_p	rate of the Perimeter Refresh Protocol
c	number of coordination points
d	threshold distance for server migration
L_p	average perimeter length
R_p	perimeter radius (protocol setting)
H	server-coordinator distance in number of hops

Table 1. Symbols

Including local communication costs around a perimeter with average length L_p, DCS/GPSR has total communication costs for requests of $\mathrm{CR}_{\mathrm{DCS}} = r_u m \cdot (\sqrt{n}/2 + L_p)$. Because perimeter forwarding may no longer be neglected, we must explicitly consider the costs of the perimeter refresh protocol, which sum up to $\mathrm{CP}_{\mathrm{DCS}} = r_p c L_p$. For DCS/BPR, each request requires $\sqrt{n}/2$ packets on average to be routed to the first relevant coordinator, and then H hops to reach the server. Thus, we have $\mathrm{CR}_{\mathrm{DCS}'} = r_u m \cdot (\sqrt{n}/2 + H)$. Server advertisements require to reach the perimeter entry node and traverse a bidirectional perimeter limited by perimeter radius R_p. Thus, we have $\mathrm{CA}_{\mathrm{DCS}'} = r_a c \cdot (H + 3R_p)$. In addition, servers must migrate their local data when they reach a critical distance to their associated coordination point. Assuming an average node speed of $\bar{v}$, we have $\mathrm{CM}_{\mathrm{DCS}'} = c \cdot (\bar{v}/d)[m/(c \cdot p)]$. Simplifying the total costs, we have:

$$T_{\mathrm{DCS}} = r_u m \left(\frac{\sqrt{n}}{2} + L_p \right) + r_p c L_p \qquad (1)$$

$$T_{\mathrm{DCS}'} = r_u m \left(\frac{\sqrt{n}}{2} + H \right) + r_a c \left(H + 3 \cdot R_p \right) + \frac{vm}{dp} \quad (2)$$

Equations 6 and 7 contain a number of free parameters. For a fair comparison of DCS/GPSR with DCS/BPR, we choose reasonable values for a MANET scenario. We assume that all data is updated with the same frequency ($r_a = r_p = r_u$), and that one server migration occurs every 10 seconds for each coordination point[1] ($v/d = 0.1$). Furthermore, we limit the average number of hops to reach a server during the second phase of routing to $H = 3$. This is consistent with the assumption that servers will be located in the vicinity of their respective coordination point. Finally, we assume a packet has a capacity of $p = 10$ data objects. The simplifications applied to Equations (1) and (2) yields:

$$T'_{\mathrm{DCS}} = r_u m \left(\frac{\sqrt{n}}{2} + L_p \right) + r_u c L_p \quad (3)$$

$$T_{\mathrm{DCS}'} = r_u m \left(\frac{\sqrt{n}}{2} + 3 \right) + r_u c \left(3 + 3 \cdot R_p \right) + 0.01m \quad (4)$$

We are particularly interested in the case where requests occur in the vicinity of coordination points, such as in [3], in contrast to a uniform distribution of requests analyzed in [13]. For the latter, we get equal curves with no significant difference. We vary the number of nodes up to $n = 1000$ and assume that all update frequencies are 5 seconds ($r_u = 5$). Figure 3 shows the total communication costs for both realizations of DCS. We can see that the average perimeter length L_p dictates the communication costs for DCS/GPSR, because *all* request packets and PRP packets make a full turn around the perimeter. This is different from BPR, where the size of the partial perimeter is always limited by the perimeter radius R_p and requests consult the first server information they find on a coordinator. Therefore, costs increase only with the number of coordination points.

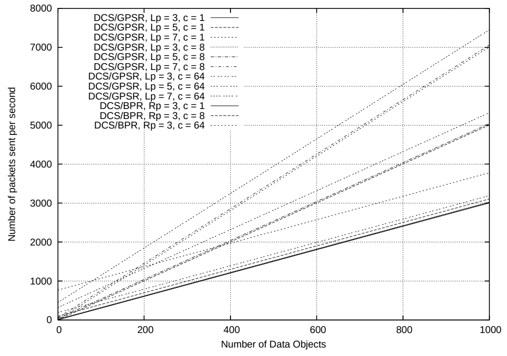

Figure 3. Communication Costs (Analytical)

[1]In reality, this value is probably large and depends on the speed and mobility model of the network.

5.2 Experimental Evaluation

We now present a set of evaluation results to compare the performance of DCS using our resilience mechanism (DCS/BPR) with DCS/GPSR as proposed in [13]. We implemented both variants of DCS in the network simulator ns-2 using its wireless extensions based on 802.11. By default, we use a total area of 600 $\times$ 600 m^2 with 150 nodes communicating over a transmission range of 100 m. All nodes move at 1.5 m/s according to the random waypoint mobility model. During 90 s of simulation, randomly selected clients issue a total of 1200 requests. We use 4 coordination points by default, each located in the center of each sub square of the area. The default perimeter radius in our experiments is 3. We consider two metrics: *communication costs* and *success rate*. Communication costs are defined as the total number of packets sent during the simulation. For both DCS/BPR and DCS/GPSR, this includes packets for requests, advertisements, and the PRP. For DCS/BPR, the success rate is defined as the fraction of requests successfully delivered to the correct server. For DCS/GPSR, the success rate is defined as the fraction of packets that could be delivered to the home node.

In Figure 4 through 6, we show the communication costs for different numbers and speeds of nodes, and number of requests, respectively. For each DCS variant, we display three settings for the number of coordination points.

In Figure 4, DCS/BPR always performs better than DCS/GPSR for the same number of coordination points. This is because DCS/BPR requires fewer packets per request than DCS/GPSR, which is due to the fact that our indirection scheme distributes server information efficiently. In contrast, DCS/GPSR requires more packets due to the relatively high number of hops required for full perimeter traversals. Note that for low node densities, DCS/BPR has *constant* costs, in contrast to DCS/GPSR, due to the fixed perimeter radius.

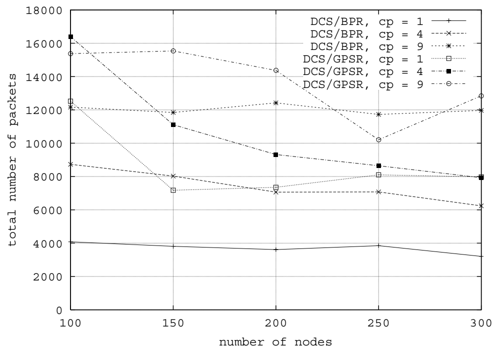

Figure 4. Communication Costs (a)

Figure 5 shows that the speed has no significant effect on the total number of packets required for both variants of DCS. However, there is a small decrease towards higher speeds, due to the fact that mobility-induced loss rate increases (compare to Figure 8). Note that speed has no significant impact on the number of packets required for our DCS mechanism, which is in accordance with our design goals.

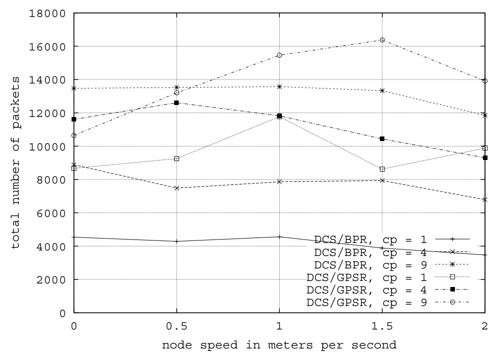

Figure 5. Communication Costs (b)

Figure 6 shows how a variation of the number of requests affects both variants of DCS. We observe that even for few requests, DCS/BPR performs better than DCS/GPSR. For an increasing number of requests, we observe the qualitative behavior that was predicted by our analytical study (Figure 3). Essentially, costs for DCS/BPR increase slower than for DCS/GPSR. The reason is that each request using DCS/BPR is able to quickly discover a coordinator, requiring fewer packets than a request travelling a full perimeter in DCS/GPSR.

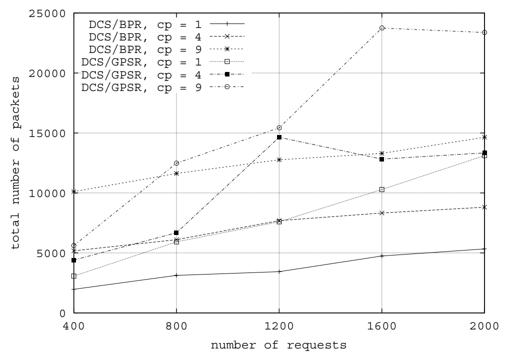

Figure 6. Communication Costs (c)

Note that in all of the previous figures, DCS/BPR requires *fewer* packets than DCS/GPSR justifying the additional indirection in our approach.

The graphs in Figure 7 through 9 show the success rate for a variation of the same parameters as in the previous graphs.

Figure 7 shows that DCS/BPR has almost constant success rate, which is better than that of DCS/GPSR. In particular, the success rate is almost unaffected by low node densities, although we only require constant communication costs, as shown in Figure 4. This is due to our advertisement strategy that effectively distributes server advertisements in the vicinity of the related coordination point at *constant* cost, because the perimeter radius is also constant. For an increasing number of coordination points, the success rate of DCS/GPSR drops well below 0.8, while the success rate of DCS/BPR is almost unaffected.

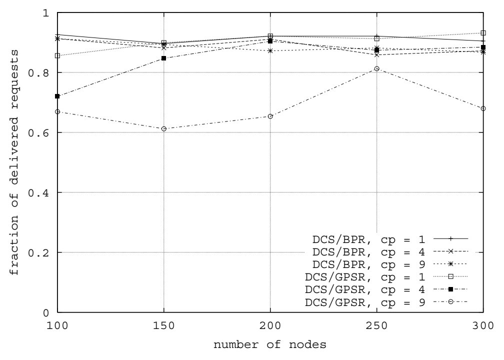

Figure 7. Success Rate (a)

Figure 8 shows that DCS/BPR has a larger success rate than DCS/GPSR also for the same number of coordination points. A small drop in the success rate can be observed, however. This is due to mobility-induced routing failures, which affects both variants of DCS to a similar extend. Note that although we use an additional indirection in forwarding requests, our success rate is *better* than that of DCS/GPSR.

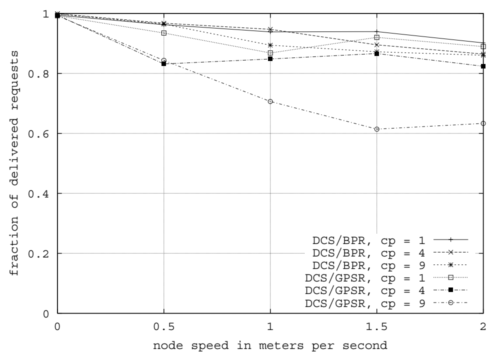

Figure 8. Success Rate (b)

The success rate for different numbers of requests is shown in Figure 9. The number of packets successfully delivered by DCS/BPR is constant across the whole spectrum and also for different numbers of coordination points. Unlike DCS/GPSR, whose success rate drops significantly when the number of coordination points increases. If more coordination points are present, some of them are located towards the border of the area. In that case, it is more likely that small closed perimeter cannot be found by DCS/GPSR. In contrast, DCS/BPR can cope well with that situation, because it uses partial perimeters.

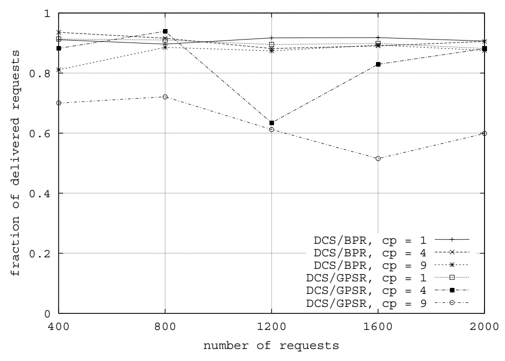

Figure 9. Success Rate (c)

6 Conclusion

This paper presented an efficient resilience scheme for operating data centric storage in MANETs. We have shown using analytical study and experimental results that over a wide spectrum of MANET scenarios, DCS using our resilience scheme outperforms DCS/GPSR as proposed in [13] both with respect to the required communication costs and the success rate of requests. Thus, our resilience scheme provides a practical an efficient solution for DCS in MANETs, which performs well for high rates of requests and a large number of data items to be managed. The latter makes it attractive for managing larger data repositories in MANETs at very low communication costs. Future work includes the design of a server migration protocol, which is able to transparently migrate the data managed by one server to another. On top of that, we plan to provide a robust resilience mechanism for data storage in MANETs that may be used by a variety of higher-level data management paradigms, such as DCS and location services.

References

[1] F. Araújo, L. Rodrigues, J. Kaiser, C. Liu, and C. Mitidieri. CHR: A distributed hash table for wireless ad hoc networks. In *Proc. of the 25th IEEE Int. Conf. on Distributed Computing Systems Workshops*, pp. 407–413, Columbus, Ohio, USA, 2005.

[2] J. Chen, Y. Guan, and U. Pooch. Customizing GPSR for wireless sensor networks. In *Proc. of the 2004 IEEE Int. Conf. on Mobile Ad-hoc and Sensor Systems*, pp. 549–551, Fort Lauderdale, Florida, USA, 2004.

[3] D. Dudkowski, P. J. Marrón, and K. Rothermel. Efficient algorithms for probabilistic queries in mobile ad hoc networks. To appear in *Proc. of the 1st Int. Conf. on Comm. System Software and Middleware*, New Delhi, India, 2006.

[4] P. Enge and P. Misra. Scanning the issue/technology - special issue on global positioning system. *Proc. of the IEEE*, 87(1):3–15, 1999.

[5] A. Ghose, J. Groklags, and J. Chuang. Resilient data-centric storage in wireless ad-hoc sensor networks. In *Proc. of the 4th Int. Conf. on Mobile Data Management*, pp. 45–62, Melbourne, Australia, 2003.

[6] Q. Huang, C. Lu, and G.-C. Roman. Reliable mobicast via face-aware routing. In *Proc. of the 23rd Annual Joint Conf. of the IEEE Computer and Communications Societies*, volume 3, pp. 2108–2118, Hong Kong, China, 2004.

[7] B. Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proc. of the 6th Annual Int. Conf. on Mobile Computing and Networking*, pp. 243–254, Boston, Massachusetts, USA, 2000.

[8] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker. Geographic routing made practical. In *Proc. of the 2nd Symp. on Networked Systems Design & Implementation*, Boston, Massachusetts, USA, may 2005.

[9] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric ad-hoc routing: Of theory and practice. In *Proc. of the Twenty-Second Annual Symp. on Principles of Distributed Computing*, pp. 63–72, Boston, Mass., USA, 2003.

[10] F. Kuhn, R. Wattenhofer, and A. Zollinger. Asymptotically optimal geometric mobile ad-hoc routing. In *Proc. of the 6th Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pp. 24–33, New York, New York, USA, 2002.

[11] X. Li, Y. J. Kim, R. Govindan, and W. Hong. Multi-dimensional range queries in sensor networks. In *Proc. of the 1st Int. Conf. on Embedded Networked Sensor Systems*, pp. 63–75, Los Angeles, California, USA, 2003.

[12] A. Rao, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In *Proc. of the 9th Annual Int. Conf. on Mobile Computing and Networking*, pp. 96–108, San Diego, California, USA, 2003.

[13] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. GHT: A geographic hash table for data-centric storage. In *Proc. of the 1st ACM Int. Workshop on Wireless Sensor Networks and Applications*, pp. 78–87, Atlanta, Georgia, USA, 2002.

[14] K. Seada and A. Helmy. Rendezvous regions: A scalable architecture for service location and data-centric storage in large-scale wireless networks. In *Proc. of the 18th Int. Parallel and Distributed Processing Symp.*, p. 218a, Santa Fe, New Mexico, USA, 2004.

[15] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin. Data-centric storage in sensornets. *ACM SIGCOMM Computer Communications Review*, 33(1):137–142, 2003.

[16] R. Tamishetty, L. H. Ngoh, and P. H. Keng. An efficient resiliency scheme for data centric storage in wireless sensor networks. In *Proc. of the IEEE 60th Vehicular Technology Conf.*, vol. 4, pp. 2936–2940, Los Angeles, CA, USA, 2004.