Infrastructureless Storage in Dynamic Environments

Jose Luis Fernandez-Marquez
IIIA-CSIC
Campus de la UAB, E-08193
Bellaterra, Spain
fernandez@iiia.csic.es

Josep Lluis Arcos
IIIA-CSIC
Campus de la UAB, E-08193
Bellaterra, Spain
arcos@iiia.csic.es

Giovanna Di Marzo Serugendo
Birkbeck College
University of London
London, UK
dimarzo@dcs.bbk.ac.uk

ABSTRACT

This paper studies the use of highly dynamic networks as infrastructures for persistent storage of data that offer services at specific geographical zones in a decentralized and distributed way. We propose a new algorithm, based on repulsion techniques, to self-organize the nodes that store and serve the information. In this work, we focus on the evaluation of our algorithm when faced to different simulated failures in order to measure its robustness and compare it with an existing approach.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]; C.2.3 [**Microcomputers**]: Portable devices; B.3.0 [**Memory Structures**]

General Terms

Algorithms, Performance

Keywords

mobile code, spatial computing, data dissemination, wireless network

1. INTRODUCTION

Nowadays, the number of connected computing devices, such as, PDAs, laptops, mobile phones, GPSs, etc are increasing continuously. They form mobile ad-doc networks (MANET's), however they are not widely exploited yet. One of the main challenges is to use these new highly dynamic environments as an infrastructure to store and offer services for the users without any centralized entity. But how to find the best location for storing or how to decide which nodes must become servers in a network topology that is changing continuously is still a challenge.

In this paper we assume that the set of mobile nodes is the environment and the pieces of information, stored in the nodes, decide on their own when and how to move among the nodes or when to self-replicate (spread), as in the *Hovering Information* problem proposed by [9]. In the hovering information problem, the information is attached to a geographical area, called anchor area. A piece of information is responsible for keeping itself alive, accessible to other devices within its anchor area by providing information to the nodes located in its communication range. Hovering information uses mechanisms such as active hopping, replication and dissemination among mobile nodes to satisfy the above requirements. It does not rely on any central server. The appealing characteristics of hovering information is the absence of a centralised entity and the active participation of the information in the storage and retrieval process.

Our work provided a series of algorithms, based on repulsion techniques, to self-organize the pieces of information that act as servers in a mobile ad-hoc network, in such a way that they ensure the maximum accessibility for the nodes that are inside the geographical area using the minimum number of nodes to store the information and minimizing the number of messages sent (information moves). In this paper we propose a new algorithm, based on repulsion techniques, and report its performance on the recovery capability when faced to failure of nodes.

The rest of the paper is organized as follows: Section 2 reports on related work. Section 3 presents the hovering information concepts and the storage areas considered. The proposed algorithm is described in Section 4. In Section 5, different types of failures are introduced and the robustness of the algorithm is studied. The paper concludes with final considerations and pointing out to future work in Section 6.

2. RELATED WORK

Hovering information is related to different concepts, such as memory, middleware or dissemination of data. In the *location-based publish/subscribe* research [6, 3] the information is moved by mobile devices and provided to mobile users according to their location. There is no need for a fixed infrastructure (no central server) but the information plays a passive role. In the hovering information approach, the information is the active entity, aware of its location, exploiting self-organisation to get stored among available devices at some specific location.

Existing *dissemination server* techniques provide alternative algorithms for spreading the information like opportunistic spatio-temporal dissemination over MANETs [7]. Existing research focuses on car traffic applications, epidemic models, such as Epcast [8] or Gossip models [2]. These works provide an interesting starting point for replication

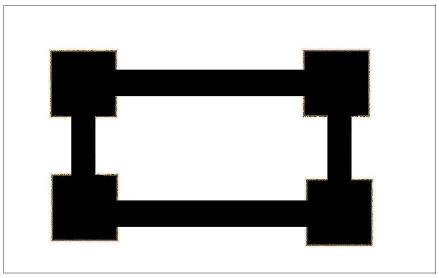

Figure 1: Amorphous Area Example

algorithms, but do not offer a solution for ensuring the persistency of the information.

The Hovering Data Clouds (HDC) concept [4], which is part of the AutoNomos project, is applied to the specific design of a distributed infrastructure-free car traffic congestion information system. Although HDCs are defined as information entities having properties similar to hovering information, the described algorithms do not consider them as an independent service but as part of the traffic congestion algorithms. The hovering information dissemination service is thought as a service independent from the applications using it.

3. HOVERING INFORMATION CONCEPT

A piece of Hovering Information h is a geo-localized information, residing in a highly dynamic environment such as a mobile ad-hoc network. A set of replicas of a piece of hovering information h are the copies created in order to ensure a good accessibility within the amorphous area. Every node that has a replica within its communication range has access to the information. The main goal of h is keeping itself alive, with the minimum cost and accessible to the mobile nodes within its amorphous area. Hovering Information uses mechanisms such as active hopping, replication and dissemination among mobile nodes to remain in its anchor area. Existing works study circular anchor areas [9]. In this paper we consider amorphous areas of any shape.

Amorphous areas are anchor areas that do not have a regular shape like a circle or a rectangle and a central point. Similarly to circular areas, the goal of a piece of hovering information is to spread itself in the area in order to be accessible to nodes in that area. Figure 1 represents an anchor area with 4 halls connected by 4 corridors. Nodes can move freely in whatever direction (also outside the corridors), i.e. the hall and corridors are not delimited by walls. The information must fill the anchor area, remain located inside it and must not spread outside. In the figure, the black color represents the positions inside the anchor area, while white color represents positions outside the anchor area. As it was presented in [9], algorithms such as broadcast and attractor point can be used to fill a circular area. Nevertheless, an approach such as Attractor point cannot be used in amorphous areas because it requires a measure of the distance between the position of the node and the center of the area. The main difference between circular and amorphous areas is that amorphous areas require of a spread mechanism to be filled.

A piece of hovering information is aware of its position in the environment and also is able to distinguish if it is inside the anchor area. Thus, any piece of hovering information carries the information of its amorphous area. We assume that the pieces of hovering information have the following information at any point in time t: (1) Knowledge of the anchor area (amorphous area in this case); (2) Position of the node it is currently in; and (3) Position of neighbouring nodes and which of them has the information already. We make also the assumption that the information itself is more expensive to spread around than getting information about position and data ids stored by neighbouring nodes.

Algorithm 1 Broadcast with Repulsion Replication Algorithm

```
pos ← NodePosition()
neigbourNodes ← NodeNeigbours()
if (InAnchorArea(pos)) then
    if (ExistsReplica(neigbourNodes)) then
        Repulsion()
    else
        Broadcast()
    end
else
    Clean()
end
```

4. BROADCAST REPULSION

In this paper we propose the Broadcast with Repulsion Replication Algorithm (BRRA) as an extension of the existing Broadcast Replication Algorithm (BRA) presented in [9]. BRA was presented for circular anchor areas. The policy followed by BRA is to trigger the replication whenever the node that stores the piece of information is inside the anchor area. This is different from the original BRA presented in [9] where the pieces of information trigger the replication when they are in a subarea called risk area inside the anchor area. It is not possible to apply this policy in the amorphous area (with an arbitrary shape), because the replicas do not spread along the desired shape when the concept of risk area is applied to amorphous shapes. The BRA replicates to all nodes in the communication range. Finally, a piece of hovering information located outside of the anchor area removes itself and frees the memory of the node in which it was located.

The goal of the Broadcast with Repulsion Replication Algorithm is to reduce the number of replicas while keeping high levels of accessibility of the information. To that purpose, BRRA: 1) replicates the information only when there is no other replica within communication range and; 2) uses a repulsion mechanism in order to spread the information over the amorphous shape. Through repulsion, the pieces of hovering information are provided with a distributed mechanism that tries to cover a maximum of the anchor area with a minimum of replicated pieces of information.

BRRA (see Algorithm 1) uses broadcast as a replication algorithm (like BRA) and repulsion to spread away the replicas. When the broadcast is executed all the nodes that are within communication range receive a replica and then they serve the information to the nodes in communication range. First, the location (inside or outside) of the replica is checked. If it is inside the anchor area, two different policies may be applied: repulsion or broadcast. If there is another information replica in a node within communication range, the repulsion is triggered. If it is the unique replica in the communication range, the broadcast is triggered. Finally, when the node holding the replica is outside the anchor area, the information is erased from the node.

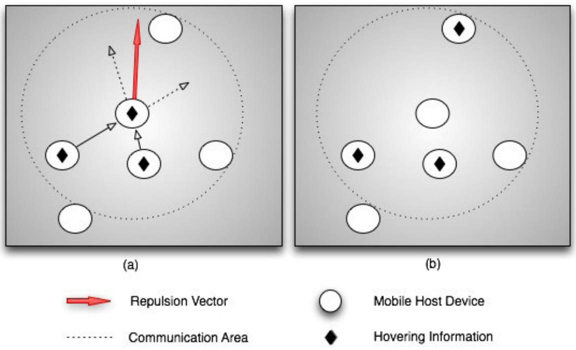

Figure 2: Repulsion example

4.1 Repulsion

The goal of the repulsion mechanism is to help spreading the pieces of information over the anchor area maintaining good accessibility levels while keeping a minimum number of replicas (in order to use less memory). When two or more replicas are close to each other, one of them will move away (removing itself from its current location and replicating further away). The repulsion mechanism, inspired by the gas theory, has been used in self-repairing formation for swarms of mobile agents [1] and as an exploration mechanism in multi-swarm optimization algorithms [5]. The main difference between our system and the use in self-repairing formation for swarms of mobile agents is that pieces of hovering information do not have any control over the movements of the mobile nodes where they reside. Even more, they do not have any knowledge about the next movements.

Let h be a piece of information, r a replica of h, and n_r the mobile node where r is currently located. The desired position for r at time $t+1$, $\vec{P}(r)_{t+1}$ is calculated as follows:

$$\vec{P}(r)_{t+1} = \vec{P}(r)_t + \vec{R}(r)_t, \qquad (1)$$

where $\vec{P}(r)_t$ is the position of r at time t and $\vec{R}(r)_t$ is the repulsion vector at time t. Next, the repulsion vector is calculated as follows:

$$\vec{R}(r)_t = \sum_{i \in R(r,t)} \frac{\vec{P}(r)_t - \vec{P}(i)_t}{dist(r,i)} \cdot (CR - dist(r,i)) \qquad (2)$$

where $R(r,t)$ is the set of replicas of h in the comunication range of n_r at time t; $dist(r,i)$ is the Euclidean distance between replicas r and i; and CR is the communication range.

Once the desired position $\vec{P_d}(r)_{t+1}$ is known, the replica r must choose which node in its communication range is the closest to this desired position. If the closest node is itself, the repulsion is not applied. Otherwise, r replicates to a new node and deletes itself from n_r.

Figure 2(a) illustrates how repulsion vectors are calculated by a replica. Repulsion vectors are inversely proportional to the distance between the position of the replica and its neighboring replicas. Using the repulsion vector, the new desired position is calculated. Next, the replica moves subsequently to the nearest node to the desired position, as it is shown in Figure 2(b). Contrarily to BRA, after applying the repulsion mechanism, the information is removed from the original node.

Blackboard	1200m x 700m
Mobility Model	Random Way Point
Number of nodes	1000
Speed of nodes	1m/s to 2 m/s
Communication Range (CR)	40 m
Algorithm Triggering	1 s

Table 1: Scenario settings

5. EXPERIMENTS

The goal of the experiments is to analyze the robustness of the proposed repulsion algorithm. Since the new algorithm reduces the number of replicas in the anchor area, we are interested in its robustness (compared with the plain broadcast algorithm) when different node failures occur.

We investigated two different node failures: (1) when the mobile nodes lose the piece of hovering information (but nodes are still available in the environment) and (2) when mobile nodes disappear (they are no longer available). Additionally, we analyze the performance of the two algorithms when the failure is uniformly distributed in the anchor area or when it is concentrated in a specific region. To measure the robustness, we analyzed the trade off between the accessibility of the information and the number of nodes with stored replicas. Three different metrics were used:

- **Accessibility**: the area covered by replicas at time t. A value of 1 represents that the 100% of the shape is covered, and 0 when there is no piece of information in the shape.
- **Memory**: the number of nodes that are storing a piece of information at time t.
- **Messages**: the number of messages sent throughout the network between time 0 and time t.

Table 1 summarises the parameter settings for the simulated scenarios. Both BRA and BBRA algorithms are triggered every 1s of simulation time.. Each run spent 270 simulation seconds and failures are simulated after 150 seconds. For each experiment, we executed 50 runs. The results presented are the average over these 50 runs.

For simulating failures in a specific region of the amorphous area, a rectangular area of $(200, 70, 800, 300)$ is selected. This rectangular area covers the left-top part of the amorphous shape (one hall and part of connecting corridors).

5.1 Initial Convergence

Previously to analyse the performance of the proposed algorithm when failures arise, we analyzed the initial convergence. We considered that the system has converged at time t, when at least 90% of the amorphous area has access to the information. In this experiment we measure the accessibility, memory, and number of messages from step 0 to step 150.

Figure 3 shows how the convergence of BRA is faster than the convergence of BRRA. The result is not surprising because BRA is continuously spreading the information using broadcast. Nevertheless, the price for this faster convergence of BRA is a significant increment of the number of messages and of the memory storage. Specifically, the number of messages sent by BRA from the beginning to step 150 is about 10.000 while with BRRA less than 4.000 messages are sent. That is, BRA duplicates the number of messages required to achieve the convergence. Regarding the use of memory, BRRA achieves the convergence with 100 replicas only, whereas BRA converges with around 400 replicas and

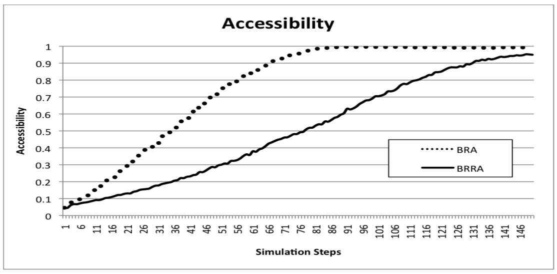

Figure 3: Initial convergence

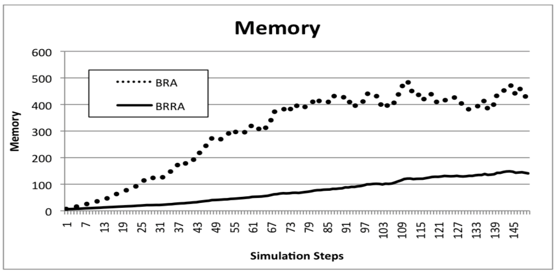

Figure 4: Memory usage in initial convergence

it uses at some points around 500 replicas (see Figure 4).

5.2 Information loss

In this scenario we simulate a failure affecting the information that nodes store, but not the network topology. That is, the information disappears in some nodes but the nodes continue moving in the system and they can be used again to store the information.

The first experiment was focused on analyzing an information failure uniformly distributed along the amorphous area. For this purpose, after 150 simulation seconds, 60% of the nodes that store a piece of information lose their information (but they continue to be present in the system). Figure 5 shows how the failure is fixed for both algorithms in a similar time. Specifically, the accessibility decreases to 0.7 after the failure but it is recovered in less than 40 simulation steps. Thus, BRRA is able to achieve the same robustness starting with a lower memory storage. Moreover, when the convergence is achieved both algorithms present a memory storage equivalent to their initial convergence (see previous experiment).

The second experiment was focused on analyzing the robustness when all the nodes located inside a specific area lose the information that is stored in their memories. The failure was also generated after 150th simulation steps. Analyzing the results (see Figure 6), we observe that the convergence of both algorithms is analogous to their initial convergence. That is, BRRA uses more time to fix the failure but the resources BRRA uses (memory and messages) is significantly lower than BRA. The use of memory decays initially in both algorithms, but recovers previous values at convergence.

5.3 Nodes Fail

In this second scenario the nodes themselves fail (disappear). This failure modifies the network topology and the number of nodes active in the system. As in the previous scenario, the first experiments were focused on a randomly distributed failure. Specifically, after 150 simulation seconds a percentage of nodes storing a piece of information

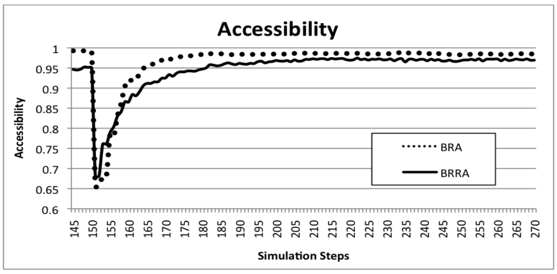

Figure 5: Convergence in random inf. loss

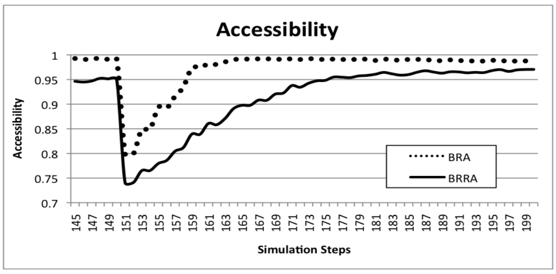

Figure 6: Convergence in area information loss

fail, and disappear from the system. Two failures were simulated: one failure involving 30% of the nodes and another failure involving 60%.

Figure 7 shows that a failure of 30% of nodes is not enough to produce a significant decrease on the accessibility of the system. The use of memory, after recovering from the failure, is the same for BRRA. This is an interesting scalability property of BRRA: the number of replicas is independent of the number or nodes. That is, it is only dependent on the number of nodes required to cover the area. On the other side, the use of memory in BRA decreases but this decrement is directly related to the decrement of nodes in the system

When the percentage of nodes that fail is increased, we observe that BRA suffers a decrease in the accessibility rates (see Figure 8). The reason for this behavior is that BRA requires a higher number of nodes to cover the area. Thus, when faced with a failure of 60% of nodes, BRA requires an additional effort to fill up the amorphous area again. On the other side, the failure of 60% of nodes in BRRA is not traumatic and the algorithm can refill the amorphous area again in a shorter time than BRA. Then, BRRA is more robust in this scenario. The number of nodes that store the information decreases more in BRA (see Figure 9) due to the big amount of nodes that are removed from the system. Nevertheless, after some simulation steps, when both algorithms have fixed the failure, the memory used by BRA is higher again.

The second series of experiments focused on analyzing the robustness when the nodes that fail are concentrated in a specific region, i.e. after the failure the complete region is empty of nodes. Thus, the recovering process requires that new nodes come to populate the region of the failure.

Figure 10 shows how, when the failure occurs (second 150), the accessibility decreases for both algorithms. The time required to fix the failure for both algorithms is very similar, because when the nodes fail, a hole in the network topology is produced and the shape can not be fixed until the network itself is fixed. BRA uses a memory proportional to

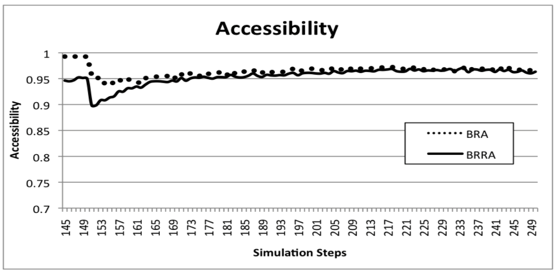

Figure 7: Convergence with 30% of random falls

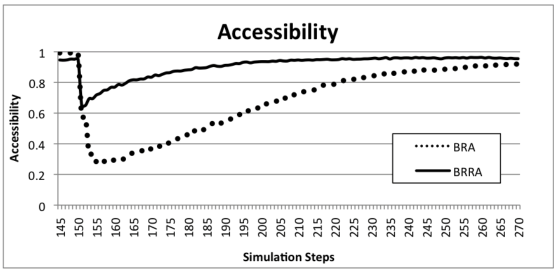

Figure 8: Convergence with 60% of random falls

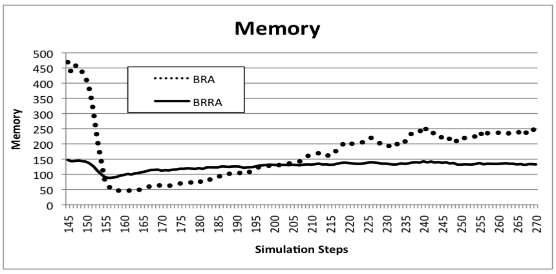

Figure 9: Memory usage with 60% of random falls

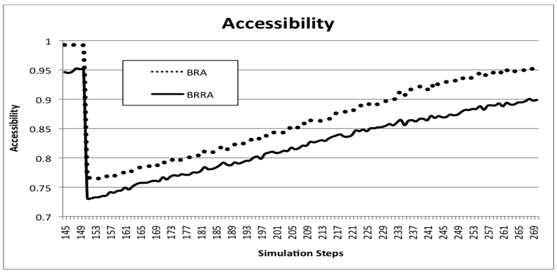

Figure 10: Convergence in an area-located failure

the number of nodes in the system. That is, the higher number of nodes the system has, the higher memory BRA uses. The number of nodes that store replicas in BRA decreases from 470 to 300, due to the lower number of nodes in the system. On the other side, after the failure, BRRA decreases the number of replicas in a smoother way and achieves the convergence without increasing the memory.

6. CONCLUSIONS

In this paper we have proposed an algorithm, based on the notion of repulsion, to let data spread itself in a pervasive system made of a set of nodes moving continuously over a bi-dimensional space. The goal of the algorithm is to fill an arbitrary shape in a bi-dimensional space with a low number of replicas, such that every node inside the shape can access to the information. We have simulated different types of failures (information failures and node failures) and shown the good performance of the proposed algorithm. Experiments have demonstrated the robustness of the algorithm when faced to failure and a significant gain in memory and messages consumption.

As a future work, we plan to evaluate the algorithm in a more realistic scenario, like a museum or a shopping mall, when the number of nodes is changing dynamically and the movement of the nodes do not necessarily follow the random way point algorithm.

7. ACKNOWLEDGMENTS

This work was funded by EVE (TIN2009-), IEA (TIN2006-15662-C02-01), and Agreement Technologies (CSD2007-0022) projects, and by the Generalitat de Catalunya under the grant 2009-SGR-1434. The first author holds a FPI scholarship from the Spanish Government.

8. REFERENCES

[1] J. Cheng, W. Cheng, and R. Nagpal. Robust and self-repairing formation control for swarms of mobile agents. In *Proc. of the 20th National Conference on Artificial Intelligence*, pages 59–64. AAAI Press, 2005.

[2] A. Datta, S. Quarteroni, and K. Aberer. Autonomous gossiping: A self-organizing epidemic algorithm for selective information dissemination in mobile ad-hoc networks. In *Proc. Int.l Conf. on Semantics of a Networked World*, LNCS, pages 126–143, 2004.

[3] P. Eugster, G. Garbinato, A. Holzer, and J. Luo. Effective location-based publish/subscribe in manets. In *IEEE International Conference on Pervasive Computing and Communications*, 2009.

[4] S. P. Fekete, , C. Schmidt, A. Wegener, and S. Fischer. Hovering data clouds for recognizing traffic jams. In *Proc. 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation*, pages 213–218, 2006.

[5] J. L. Fernandez-Marquez and J. L. Arcos. Keeping diversity when exploring dynamic environments. In D. Shin, editor, *Proc. 24th Annual ACM Symposium on Applied Computing*, pages 1192–1196, 2009.

[6] I. Leontiadis, P. Costa, and C. Mascolo. Persistent content-based information dissemination in hybrid vehicular networks. In *IEEE International Conference on Pervasive Computing and Communications*, 2009.

[7] I. Leontiadis and C. Mascolo. Opportunistic spatio-temporal dissemination system for vehicular networks. In *Proc. 1st Workshop on Mobile opportunistic networking*, pages 39–46, 2007.

[8] S. Scellato, C. Mascolo, M. Musolesi, and V. Latora. Epcast: Controlled dissemination in human-based wireless networks by means of epidemic spreading models. *CoRR*, abs/0711.2780, 2007.

[9] A. Villalba Castro, G. Di Marzo Serugendo, and D. Konstantas. Hovering information - self-organising information that finds its own storage. In A. Vasilakos, M. Parashar, S. Karnouskos, and W. Pedrycz, editors, *Autonomic Communications*. Springer-Verlag, 2009.