An Evaluation of Location-Demographic Replacement Policies for Zebroids

Shahram Ghandeharizadeh and Shyam Kapadia
Department of Computer Science
University of Southern California
Los Angeles, CA 90089, USA
Email:shahram,kapadia@usc.edu

Abstract

In an ad-hoc network of mobile devices, a device is termed a zebroid when it carries a data item residing on a server-device to a client-device referencing that item. The system employs a zebroid when its travel path intersects that of the server and the client and this information is known in advance. The motivation for use of a zebroid is to minimize the delay incurred by the client for the referenced data item, termed the availability latency. This paper considers the performance of alternative policies that manage the identity of data items assigned to a zebroid when its storage is exhausted. One novel policy is LoDeR that manages storage of zebroids based on the demographics of a geographical region where the zebroid rendezvous with the client. Experimental results demonstrate a host of tradeoffs between the different performance metrics such as the number of data items lost by a policy and the percentage of requests that reference these data items, availability latency, and number of replaced data items. The mobility model has a significant impact on these metrics for a given policy.

1 Introduction

In-vehicle entertainment systems have evolved to deliver continuous media (audio and video clips) to passengers. This marks a new trend toward 'entertainment on wheels', enabled using DVD players, fold-down screens, and wireless headsets. Vehicles may be equipped with devices consisting of hundreds of gigabytes of storage, a fast processor and a wireless interface with bandwidths in the order of 10s and 100s of Megabits per second. These wireless interfaces following standards such as the IEEE 802.11a/g have a limited radio range, typically in the order of a few hundred feet [3]. The high bandwidth peer-to-peer network formed between the vehicles constitutes the data plane of these so called 'C2P2' devices [7]. C2P2 enabled vehicles may collaborate to form a mobile ad-hoc network that delivers a requested data item from a server to a client.

Without loss of generality and to simply discussion, in the rest of this paper, the term car refers to a vehicle with a C2P2 device. We assume a two-tiered networking infrastructure consisting of a high bandwidth data plane and a low bandwidth control plane. The data plane is the ad-hoc network described in the previous paragraph. The control plane is a connection to the wired infrastructure that is low bandwidth with high reliability. Examples include a reliable cellular connection between a vehicle and a cellular base station offering bandwidths in the range of Kilobits per second. The control plane is essential for a car to transmit service oriented data such as billing information and other useful information such as the current position of a car, its destination and travel path.

Mobility offers the opportunity for cars to act as data carriers. When a client references a clip (say audio clip X), a car whose path intersects the client may carry clip X to it. These collaborative cars are termed 'zebroids'. The concept of zebroids is made possible by in-vehicle navigation systems which compute the path of travel given a destination. The control plane is employed to communicate this critical information to a centralized dispatcher.

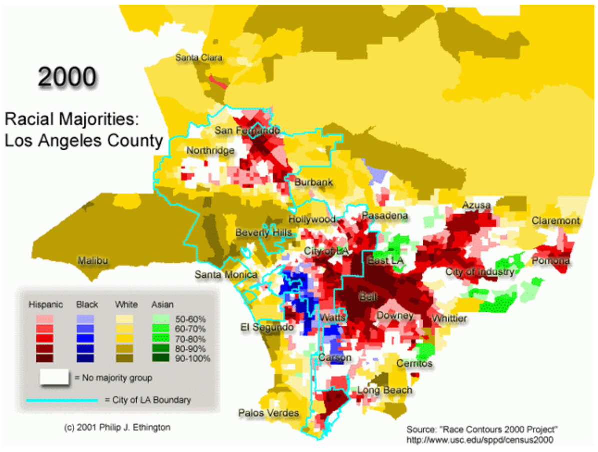


Figure 1: Demographics of the Los Angeles county.

In a typical scenario, the interface of a C2P2 device provides an in-vehicle user with a list of available data items and the incurred delay before the client encounters a copy of each presented data item. Here, the term data item refers to all data types including audio and video clips, etc. Availability latency, δ, is the delay from when a user references a data item to when one of its replicas become available. This latency is a function of the following parameters: (i) current location of the client, (ii) demographics of this location and how well it matches the assumed demographics of the referenced data item, (iii) destination and travel path of the client, (iv) mobility model of the C2P2 equipped cars, (v) number of replicas constructed for the referenced data item, (vi) placement of data item replicas across cars. When a replica of the referenced data item resides on the client's local storage, δ is zero.

When the client does not contain a replica of the referenced data item, the value of δ is dictated by the earliest time when a car with a replica of that data item (termed server) rendezvous with the client. The value of δ is minimized when the system identifies a zebroid to carry a replica of the referenced data item from the server to the client. The zebroid must be in close proximity of a server and have a travel path that reaches the client sooner[1]. To compensate for possible error in mobility models, the system may use multiple zebroids to carry the same data item, increasing the likelihood of rendezvous with a client.

The population of a geographical region is categorized into different demographics, dictating their choice of entertainment content. For example, Figure 1 shows the Los Angeles county and its different demographics. A Hispanic neighborhood will be more interested in Spanish titles (audio and video) when compared with a non-Spanish speaking neighborhood. Hence, a system that provides entertainment on the move to customers as they travel in their vehicles may consider the population preferences of its geographical area. We term one such area a zone. Cars located in a particular zone may strive to maintain many replicas of titles frequently accessed in that zone.

All cars including zeborids have a finite amount of storage. When the local storage of the zebroid chosen by the dispatcher is completely occupied, the zebroid must replace one or more data items to accommodate the referenced data item that it then carries to the requesting client. Various choices of replacement policies are possible. A zebroid may evict a data item randomly. Another choice is to evict an item not popular in the zone in which it will rendezvous with the client. Yet another choice is to evict an item not popular in the destination zone of the zebroid. We label the last two policies as LoDeR-Rdz and LoDeR-Dest since they use the location demographic information of the zones. The focus of this paper is to quantify the tradeoff associated with these policies. We assume the dispatcher has knowledge of mobility of each car, i.e., position of each car is known as a function of time. We made this simplifying assumption in order to minimize the number of random parameters to clearly understand and document the merits and drawbacks of alternative policies. This serves to justify other variations of LoDeR such as its decentralized implementation or

[1]This paper does not consider transitive zebroids that hand data to one another to expedite delivery of data. In addition, we ignore candidate zebroids that may rendezvous with a server at a later point in time (relative to when the request for a data item is issued) and rendezvous with the client sooner. Both constitute our future research directions.

Study	Potentially Mobile Nodes	Mobility Model	Delivery	Energy Efficiency	Optimize Delay	How many copies created?	Storage Constraint
ZebraNet [8]	All	Controlled + species dependent	Many to One	X		Many	X
DakNet [11]	One	Controlled	One to One			One	
Message Ferries [16]	All	Random + Controlled	One to One	X	X	One	
SWIM [13]	Most	Random without predictions	Many to Some		X	Many	
Data Mules [12]	Some	Random without predictions	Many to One			One	X
Seek and Focus [14]	All	Random without predictions	One to One	X	X	One	
Our Work	All	Random with predictions	Any to One		X	One or More	X

Table 1: Related DTN studies.

use of multiple zebroids to combat possible error in anticipated rendezvous times due to factors such as traffic.

The primary contribution of this paper is an evaluation of a number of replacement policies to better utilize the system storage. The study depicts a variety of tradeoffs between the policies in terms of the following performance metrics: availability latency, replacement overhead and data loss incurred by a policy. A key observation is as follows. With LoDeR, system's performance metric are impacted by the assumed mobility model.

The rest of this paper is organized as follows. Section 2 presents a brief overview of the related work. Section 3 introduces various replacement policies used in this study. Section 4 describes the results of our simulation study using the zone-based framework. Finally, Section 5 presents brief conclusions and future research directions.

2 Related Work

A large number of literature in the area of delay tolerant networks (DTN [5]) where data carriers like zebroids have been deployed. Table 1 gives a brief summary of these studies distinguished on the basis of a number of dimensions. In the following paragraphs, we give a brief description of some related studies.

With sparse ad-hoc networks, a large number of studies [10, 16, 9, 8, 11, 14, 12, 15] attempt to deliver data taking advantage of node mobility. The studies can be characterized into reactive and proactive on the basis of whether the mobility of the nodes is controlled. In the reactive case, node movements are dictated by a specific mobility model and applications rely on the movement inherent in these nodes for data delivery. In the proactive case, the node movement can be adapted proactively to deliver data and also in some cases reduce data delivery latency. We first summarize the studies that use proactive schemes followed by a brief description of ones that employ reactive schemes.

Li and Rus [10] introduce a scheme that computes the optimal trajectory for relaying a message among nodes. However, for larger networks, supporting multiple simultaneous message transmissions using this algorithm is difficult because the scheduling problem becomes intractable. Along the same lines, in [16], special nodes called message ferries that follow nonrandom movement paths allow data delivery between other nodes whose movements are governed by a random mobility model. The movement of message ferries can be controlled in order to obtain data from nearby source nodes and deliver it to the appropriate sink nodes when in vicinity. Message transmissions only take place via direct transmissions. In a follow-up work [9], latency-energy tradeoffs are demonstrated by the authors where a single ferry follows a known trajectory and the other nodes schedule their sleep/wakeup schedule in accordance with when the ferry will be in the vicinity. Comparison with reactive protocol like DSR indicate significant energy savings while suffering only a small drop in delay performance.

In the ZebraNet project [8], sensors are attached to zebras to study the wild life behavior. The sparse nature of the network prevents formation of a connected network. Hence, data is obtained from the sensors when humans drive by in a car or some other vehicle. Similarly, in the DakNet project [11], vehicles are used to transfer data between villages and cities using a store and forward mechanism. Our work with zebroids differs from all the above studies that can categorized in the proactive realm, in that, it is reactive and employs a random walk mobility model. The movement of the various nodes (vehicles in our case) is dictated by this model and cannot be controlled explicitly as was utilized in the above studies.

Nodes that obey a random mobility model have been widely studied in the context of reactive schemes. There have been a large number of studies on the analysis of properties of random walks [2] on a torus. Some recent studies like [14, 12] have used the 2D random walk mobility model in the context of routing in intermittently connected mobile networks. Our mobility model is similar to that used in these studies.

In [12], DataMules are used to route data from static sensors to the base-stations in a sparse sensor network. In our studies, all nodes are mobile and have the same capabilities. In [14], using latency or meeting time and number of transmissions as the metrics of interest, the authors present comparisons of a randomized, utility based and an hybrid (seek and focus) approach with an optimal scheme via analysis as well as simulations. The metric of meeting time used in this study is analogous to our notion of availability latency. However, our study differs from the work by Spyropoulos et. al [14], in that, their study does not consider a storage constraint per node as is the case with zebroids. The storage constraint requires zebroids to make decisions about which data item replicas should be kept in local storage to minimize overall availability latency. Moreover, we do not expect energy to be a very constrained resource in a C2P2 environment, hence we have not consider number of transmissions as a metric in our studies. Finally, their study does not employ an interference model that will constrain the available bandwidths for data transfer in a C2P2 environment.

Some studies in the mobile infostation context have relied on replication and caching techniques to reduce data delivery latency. Small and Haas [13] propose the Shared Wireless Infostation Model (SWIM) where the infostation architecture is combined with the ad-hoc networking model. Here, the infostations act as data sinks. By replicating and hence spreading data across the mobile nodes data delivery latency at the infostations can be greatly reduced. This study uses a differential equation model to analytically determine the time until the data is spread to all the nodes. Presence of static infostations, a different mobility model, an absence of a storage constraint per node and a different architectural framework and application makes this study significantly different from ours.

3 Replacement policies

This section outlines several replacement policies to enable a zebroid to choose a victim object to swap out when its storage capacity is fully occupied.

Location-Demographic Replacement policy (LoDeR) assumes each zone has a list of data items for each demographic sorted based on their frequency of access. The dispatcher identifies the zone where the zebroid rendezvous with the client (say Z_R). The victim data item is the least frequently accessed one belonging to zones other than Z_R. In other words, the least popular data item that matches demographics of Z_R is kept while the frequently accessed data item of some other zone is evicted. When all data items stored with a zebroid belong to Z_R, the dispatcher evicts the least frequently accessed one. There are two variations of this policy: LoDeR-global and LoDeR-local. We describe each in turn. With **LoDeR-global**, the dispatcher maintains the frequency of access to data items based on all requests issued by all cars. The centralized paradigm uses this information to implement LoDeR. With **LoDeR-local**, each car maintains the frequency of access to data items based on the requests issued by itself and its neighbors. This facilitates a decentralized implementation that might be deployed on each C2P2 device.

Each of the LoDeR policies above can in turn be classified on the basis of whether the zebroid makes a replacement on the basis of its rendezvous zone or its destination zone. Specifically, the two variants are: 1) Replaces content based on the rendezvous zone of the zebroid and client, termed Rdz, and 2) Replace content based on the destination of the zebroid, termed Dest.

We now present a few other replacement policies. **Random** is the simplest policy where a zebroid evicts a data item chosen using a simple random coin toss. The next policy is **Least frequently used policy (LFU)** and has two variations: (a) **Local (LFU-local):** requires each C2P2 to maintain the least frequently used data item within its local repository. During eviction, this is the candidate replica that is replaced. With **Global (LFU-global)**, the dispatcher maintains the frequency of access to the data items based on all client requests. When a zebroid contacts the dispatcher for a victim data item, the dispatcher chooses the data item with the lowest frequency (global) of access.

The replacement policies incur the following overheads. First, the complexity associated with the implementation of a policy. Second, the bandwidth used to transfer a copy of a data item from a server to the zebroid. Third, the average number of replacements incurred by the zebroids. Fourth, a zebroid may replace the only copy of a data item left in the system. Hence, replacement policies may result in data loss. This loss is prevented when the dispatcher stipulates that a victim data item is the last copy of that item and chooses another victim data item to maintain one copy of each data item in the ad-hoc network at all times.

In this study, we only present the results for the random and the global replacement policies, namely LoDeR-Dest, LoDeR-Rdz and LFU-global, noting that the results of the local variants of these policies are close to that seen with the random policy.

4 Evaluation

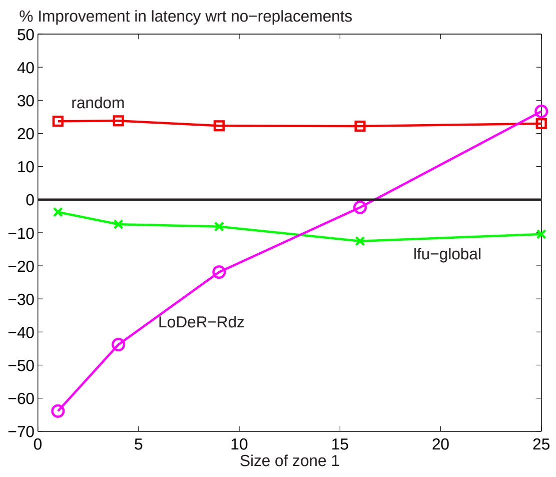
4.1 Simulation set-up

We consider a 6×6 2D torus as the map for illustration. A Markovian mobility model representing a 2D random walk on the surface of the torus describes the movement of the cars across this torus. Each grid/cell of this map is a unique state of this Markov chain. Only C2P2 equipped cars within the same cell may communicate with each other. A car transitions from a cell to any of its neighboring 8 cells. In each time slot, every car moves from one cell to another depending on its current position and probability transition matrix $Q = [q_{ij}]$ where q_{ij} is the probability of transition from state i to state j. A homogeneous repository of equi-sized data items is used ($S_i = 1$). A C2P2 device does not maintain more than one replica of a data item. This is because additional replicas occupy storage without providing benefits.

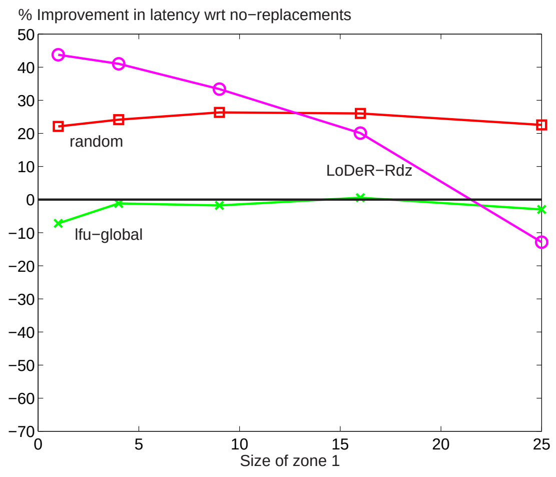
Cars move freely across the map as defined by the Markov mobility model. In our experiments, we looked at two mobility models: 1) a car moves constantly from its current position to a randomly chosen destination, termed Moving, and 2) a car moves from its current position to a randomly chosen destination and waits for the length of its trip duration before repeating the process, termed Halt. With the Halt model, to prevent synchronization such that all cars halt at the same time, we have randomized the start times of the vehicles. Future research will consider mobility models where movement of some cars may be restricted to a zone as well movement traces generated from a microscopic vehicular simulator like CORSIM [1].

The map is partitioned into two zones. The zones can be considered analogous to neighborhoods each fitting a particular population demographic. There are a total $T = 40$ data items in the system. Each zone has a list of 20 data items for which it has a preference. The preference lists of data items for the two zones are disjoint. A client in zone 1 references a data item corresponding to the demographics of that zone. Similar is the case for a client in zone 2. In each zone, requests are generated according to a Zipf distribution across the preferred 20 data items with a mean of 0.27. This distribution is shown to correspond to sale of movie theater tickets in the United States [4]. There is only one request active in the system at all times. A new request is issued alternately in zone 1 and zone 2 respectively. The choice of the car to which a new request is assigned (client) in a specific zone is uniform. Hence, in case of the Halt model, it may be the case that a request is issued at a car that is stationary.

Let α represent the number of storage slots per C2P2. Each storage slot stores one data item. γ represents the duration of the client's journey in terms of the number of time slots. In all our experiments, we set $\gamma = 10$. Availability latency (δ) is defined as the number of time slots after which a client C2P2 device will encounter a replica of the data item for



2.a) zone-1



2.b) zone-2

Figure 2: Improvement in δ_{agg} wrt the no-replacements policy as a function of the size of zone 1 with the Moving mobility model, when loss of data is not allowed. Here $N = 40$, $\alpha = 8$ and $T = 20$ per zone.

the first time. If a replica for the data item requested was encountered by the client in the first cell then we set $\delta = 0$. If $\delta > \gamma$ then we set $\delta = \gamma$ indicating that no copy of the requested data item was encountered by the client during its entire journey. We are interested in the availability latency observed across all data items. Hence, we calculate the average availability latency δ_{avg} as : $\delta_{avg} = \sum_{i=1}^{T} \frac{\delta_i}{T}$. This latency metric is calculated specific to each zone.

The number of cars distributed across the map is set to 40 in all experiments ($N = 40$). Initially the cars are distributed uniformly at random across the cells of the map. The number of replicas per data item is computed proportional to the total available storage and the square-root of the frequency of access to the data items [6]. Replicas are distributed across the cars uniformly at random.

We consider two scenarios, one in which at least one replica of every data item is maintained in the ad-hoc network at all times (*no-loss*) and the other in which there is no such restriction (*loss*). As a comparison yard-sick, we compare the alternative replacement policies with an environment that does not employ zebroids. This is termed a 'no-replacements' environment and incurs no data loss.

The metrics of interest are the aggregate availability latency (δ_{agg}), number of replacements (captures overhead of a policy) and percentage of requests to the lost data items (for the loss case). When data loss is allowed, the replacement policies may evict the only remaining replica of a data item from a zebroid causing the system to lose that item. The last metric captures the requests to such data items that are lost from the system.

All results in this section are averages across $100,000$ requests.

4.2 Results

We first present the results with the Moving mobility model, followed by a brief description of the results with the Halt mobility model.

4.2.1 Moving Mobility Model

Figure 2 presents the percentage improvement in availability latency with the alternative policies as a function of the size of zone 1. In this experiment, the dispatcher ensures that at least one copy of a title is available in the system (this assumption is removed in the results of Figure 3). The x-axis of this figure denotes the number of cells occupied by zone 1. We partition the 36 cells in the map into two zones. The shape of zone 1 is a square grid located at the center of this 6×6 torus. We consider 5 different sizes for zone 1 representing a $m \times m$ square grid where $m = \{1, 2, 3, 4, 5\}$. Note that if size of zone $1 = x$ then size of zone $2 = 36 - x$, $0 \leq x \leq 36$. The shape of zone 2 is different because its cells surround zone 1.

The y-axis of Figure 2 is the percentage improvement in availability latency when compared with the no-replacement scenario. If the average availability latency with a policy is p and that of no-replacements scenario is nr, the percentage difference is defined as $\frac{p-nr}{nr} \cdot 100$. When the percentage difference is negative, it means the no-replacements scenario provided superior availability latency. Figure 2.a and 2.b show the results for requests issued by cars in zone 1 and zone 2 respectively. We eliminate LoDeR-Dest from the results since its obtained results are identical to LodeR-Rdz. This is because with the Moving mobility model a car does not halt once it arrives at its destination.

Figure 2 shows the random policy is robust, providing a superior availability latency that is consistently better than the no-replacements environment. There are several reasons for this. First, the no-replacements environment does not use zebroids when their storage capacity is exhausted while the random policy employs zebroids. Second, the mobility pattern used here is random, hence the locations from which the requests are issued by the client is also random. Moreover, the replacement decisions made by a zebroid are blind to future requests issued by the cars. (Note that a zebroid might be either a client or a server for future requests.) These findings are significant because a random policy can be implemented in a simple, decentralized manner.

Policies such as LFU-global and LoDeR-Rdz are more interesting because the centralized dispatcher has a profile that better approximates the distribution of requests to the different data items. Both LFU-global and LoDeR-Rdz show they are sensitive to the shape and structure of zones. This trend is magnified with LoDeR-Rdz. In the following, we explain the behavior of these two techniques in turn.

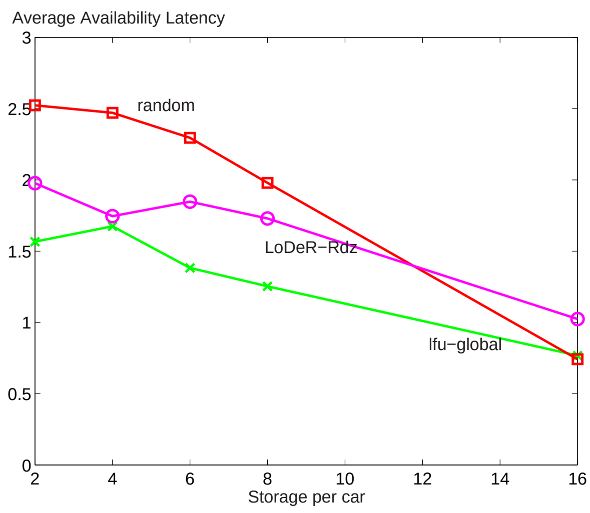
The LFU-global policy performs worse than the no-replacements case for both zone 1 and zone 2. Since it is global, it rapidly develops a preference for the more popular data items creating a large number of replicas for them. This is at the expense of the less popular items and for the least popular ones it just maintains the minimum of 1 replica in the system. The higher number of replicas provide almost no additional benefits, while the lower number of replicas for the other data items hurts the latency performance of this policy. The no-replacements scenario employs the square-root technique for initial placement of data [6] which results in a better allocation of storage.

LoDeR-Rdz improves the availability latency for the large zones. For example, when zone 1 is less than 16 cells, availability latency incurred for data items belonging to this zone is worse than the no-replacements scenario (see Figure 2.a). Zone 2 which is larger observes more than 15% improvement in its incurred availability latency (see Figure 2.b). Note that Figures 2.a and 2.b are not symmetrical because the shape of the two zones is different: zone 1 corresponds to a $m \times m$ grid at the center of the 6x6 torus while zone 2 is composed of the cells surrounding zone 1.

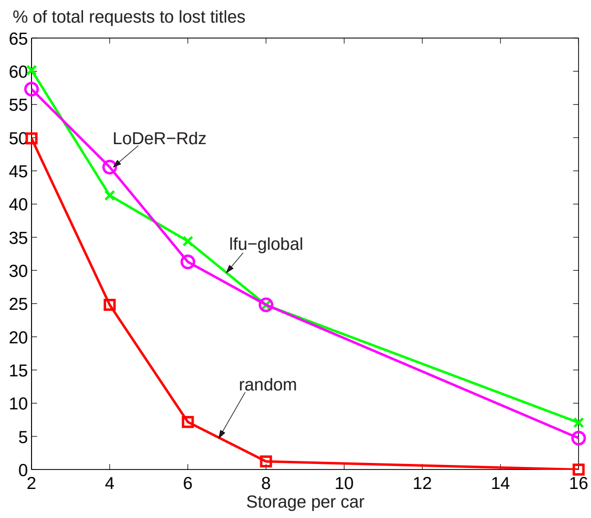
Next, we analyze the impact of data loss and the overhead of different techniques. In these experiments, we maintain a constant value for N, T, and the size of zone 1 (40, 20, and 9 respectively) while varying $\frac{S_T}{S_{DB}}$ ratio. Obtained results are shown in Figure 3. The x-axis of these figures shows the different $\frac{S_T}{S_{DB}}$ ratios with a larger value reflecting a greater amount of storage relative to the database size. Note that $S_T = N \cdot \alpha$ while $S_{DB} = T$.

Figure 3.a shows LFU-global and LoDeR-Rdz policies provide a better availability latency when the amount of storage is scarce, i.e., $\frac{S_T}{S_{DB}}$ is 2. When storage is abundant, i.e., $\frac{S_T}{S_{DB}}$ approaches 16, all policies provide almost the similar availability latency. When storage is scarce, LFU-global and LoDeR-Rdz lose the less popular data items in favor of the most frequently accessed ones (see Figure 3.b). Hence, the number of servers for the popular data items in the system increases, reducing the role of zebroids in minimizing availability latency[2]. This is shown in Figure 3.c where

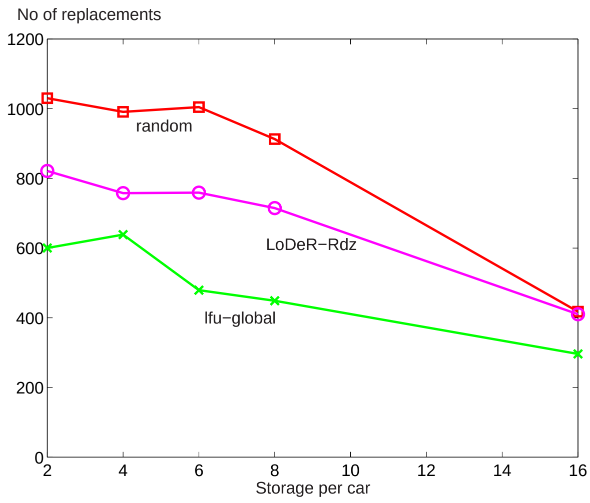
[2]This does not contradict the explanation provided for poor performance of LFU-global in Figure 2 because it did not allow for loss of data



3.a) δ_{agg}

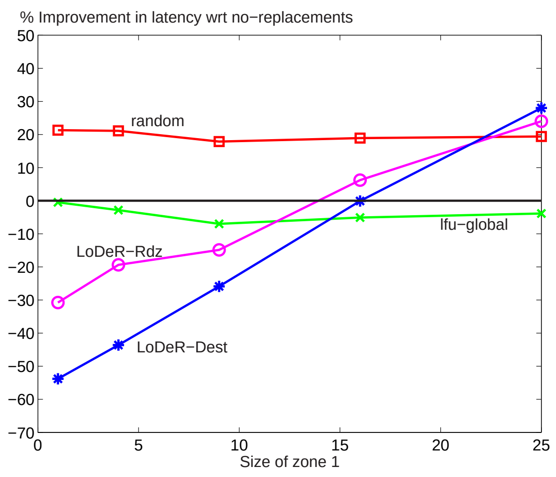


3.b) Lost Data item requests

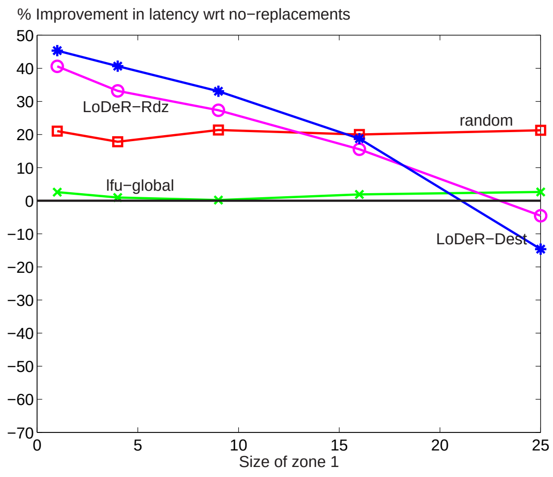


3.c) No of replacements

Figure 3: Performance of the different replacement policies in zone 1 as a function of the S_T/S_{DB} ratio with the Moving mobility model, when loss of data is allowed. Here $N = 40$, size of zone 1=9 and $T = 20$ per zone.



4.a) zone-1



4.b) zone-2

Figure 4: Improvement in δ_{agg} wrt the no-replacements policy as a function of the size of zone 1 with the Halt mobility model, when loss of data is not allowed. Here $N = 40$, $\alpha = 8$ and $T = 20$ per zone.

the number of replacements is lower for LoDeR-Rdz and LFU-global when storage is scarce. On the other hand, the random policy exhibits a lower data loss but at the cost of a higher latency and higher replacement overhead. Hence, depending on which metric is important for an application environment an appropriate policy may be selected.

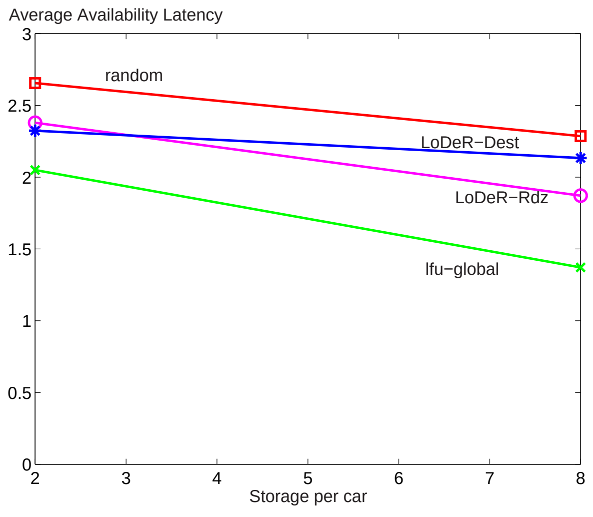
4.2.2 Halt Mobility Model

Figures 4.a and 4.b show the results for percentage improvement in average availability latency as a function of the size of zone 1, for requests issued by cars in zone 1 and zone 2 respectively, when no data loss is allowed. These results show LoDeR-Dest sacrifices the average availability latency observed by the small zone by a significant amount for modest enhancements in the availability latency observed by the larger zone (see Figure 4). In the following, we explain this observation.

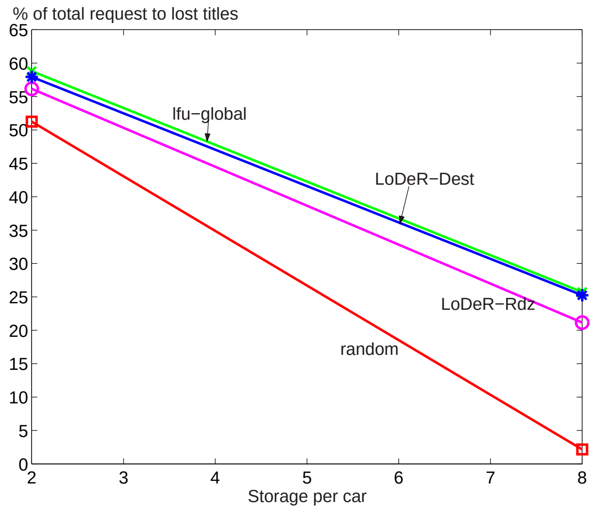
Let $P_{Halt,Rdz}$ denote the probability of replacing data items of the smaller zone with LoDeR-Rdz. Let $P_{Halt,Dest}$ denote the probability of replacing data items of the smaller zone with LoDeR-Dest. Our key observation is that $P_{Halt,Rdz}$ is smaller than $P_{Halt,Dest}$ for the small zone. This is because a car is less likely to stop in a small zone as compared to the likelihood of passing through that zone. This means the LoDeR-Dest policy swaps-out a larger number of data items popular with small zones, reducing the total number of data item replicas desired by this zone. Availability latency observed by a zone is dictated by the number of replicas of its referenced data item. When comparing LoDeR-Rdz with LoDeR-Dest, a higher $P_{Halt,Rdz}$ results in a higher availability latency for the small zone. At the same time, this does not translate into significant improvements for the availability latency of titles assigned to the larger zone. This is because, with Halt, the zebroid may stop at any cell of the larger zone and not encounter a candidate client requesting data items within this zone. With the Moving mobility model, the two alternative policies provided an identical behavior because $P_{Halt,Rdz}$ is the same as $P_{Halt,Dest}$.

Figure 5 shows the latency, replacement overhead and percentage of lost title requests for the various policies with the Halt model when data loss is allowed. The trends seen with the behavior of the relative performance of the LoDeR policies versus lfu-global and random are identical, the same set of lessons and explanations as were described in the case of the Moving model apply here. The only difference is the relative performance of the LoDeR-Rdz and LoDeR-Dest policies. It turns out that when data loss is not allowed LoDeR-Rdz sacrifices a lower number of data items for the smaller zone as compared to LoDeR-Dest. The explanation for this is identical to that provided for Figure 4.

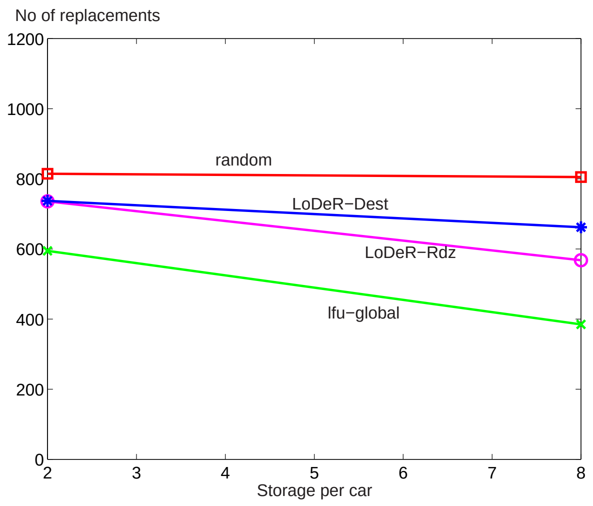
items.



5.a) δ_{agg}



5.b) Lost Data item requests



5.c) No of replacements

Figure 5: Performance of the different replacement policies in zone 1 as a function of the S_T/S_{DB} ratio with the Halt mobility model, when loss of data is allowed. Here $N = 40$, size of zone 1=9 and $T = 20$ per zone.

5 Conclusions

LoDeR replacement policies manage the contents of a zebroid based on the demographics of the location which serves as the rendezvous point with a client or is the destination for the zebroid. The primary motivation for such a policy is to increase the likelihood of the zebroid becoming a server for other cars in the vicinity of its rendezvous point, minimizing the incurred availability latency.

Obtained experimental results show that availability latency of a data item is impacted by the number of replicas of a data item, which in turn depends on the mobility model and the replacement policy employed. With LoDeR policies, we observe modest improvements in latency with both a Halt and Moving model for large zones at the expense of a higher availability latency for data items belonging to the demographics of a smaller zone. When loss is allowed, LoDeR policies lose less popular data items belonging to different zones in favor of additional replicas for the popular data items. Whether this is acceptable is application dependent. For example, it might be acceptable to lose commercials targeted towards a small minority group occupying a small geographical region in favor of a lower availability latency for a larger population occupying a large geographical area.

An important finding of this study is that a simple random replacement policy provides competitive performance under both the Halt and Moving mobility models. This is because the simulations assume a random walk mobility model where future requests are not known in advance.

Our future research directions are to investigate alternative variations of the LoDeR policies with different mobility models. Extensions that consider the use of transitive zebroids where a data item may be handed over from one zebroid to another over time may provide further latency improvements. We intend to consider this and use of multiple zebroids that employ LoDeR policies as part of our future work.

6 Acknowledgments

We wish to thank Bhaskar Krishnamachari for his valuable comments and suggestions. This research was made possible by an unrestricted cash gift from Microsoft research, a fellowship award from the Annenberg Center for Communication, and NSF research grant IIS-0307908.

References

[1] Federal Highway Administration. Corridor simulation (corsim/tsis). Version 5.1, http://www.ops.fhwa.dot.gov/trafficanalysistools/corsim.htm.

[2] D. Aldous and J. Fill. Reversible markov chains and random walks on graphs. Manuscript under preparation, http://stat.www.berkeley.edu/users/aldous/RWG/book.html.

[3] S. Bararia, S. Ghandeharizadeh, and S. Kapadia. Evaluation of 802.11a for Streaming Data in Ad-hoc Networks. In *ASWN Boston, MA*, 2004.

[4] A. Dan, D. Dias, R. Mukherjee, D. Sitaram, and R. Tewari. Buffering and Caching in Large-Scale Video Servers. In *COMPCON*, 1995.

[5] K. Fall. A delay-tolerant network architecture for challenged internets. In *SIGCOMM ’03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 27–34, New York, NY, USA, 2003. ACM Press.

[6] S. Ghandeharizadeh, S. Kapadia, and B. Krishnamachari. Comparison of Replication Strategies for Content Availability in C2P2 networks. In *6th International Workshop on Mobile Data Management*, May 2005.

[7] S. Ghandeharizadeh and B. Krishnamachari. C2P2: A Peer-to-Peer Network for On-demand Automobile Information Services. In *GLOBE*, Aug 2004.

[8] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet. *SIGARCH Comput. Archit. News*, 30(5):96–107, 2002.

[9] H. Jun, W. Zhao, M. Ammar, C. Lee, and E. Zegura. Trading latency for energy in wireless ad hoc networks using message ferrying. In *Third IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'05)*, 2005.

[10] Q. Li and D. Rus. Sending messages to mobile users in disconnected ad-hoc wireless networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 44–55, New York, NY, USA, 2000. ACM Press.

[11] A. Pentland, R. Fletcher, and A. Hasson. DakNet: Rethinking Connectivity in Developing Nations. *Computer*, 37(1):78–83, 2004.

[12] R. Shah, S. Roy, S. Jain, and W. Brunette. Data Mules: Modeling and Analysis of a Three-Tier Architecture for Sparse Sensor Networks. *Elsevier Ad Hoc Networks Journal*, 1, September 2003.

[13] T. Small and Z. J. Haas. The shared wireless infostation model: a new ad hoc networking paradigm (or where there is a whale, there is a way). In *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 233–244, New York, NY, USA, 2003. ACM Press.

[14] T. Spyropoulos, K. Psounis, and C. Raghavendra. Single-Copy Routing in Intermittently Connected Mobile Networks. In *SECON*, April 2004.

[15] T. Spyropoulos, K. Psounis, and C. Raghavendra. Spray and wait: An efficient routing scheme for intermittently connected mobile networks. In *WDTN Workshop in association with ACM SIGCOMM*, 2005.

[16] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *MobiHoc '04: Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, pages 187–198, New York, NY, USA, 2004. ACM Press.