Data Management in Mobile Networks

CMSC 699 Independent Study Report

Palanivel Kodeswaran

palanik1@cs.umbc.edu

1 Introduction

The improvements in the computational and communication capabilities of mobile devices has made them all pervasive and is leading to Ubiquitous computing in which all computations will fade into the background. Much of research has gone into improving the functionality, computational power, size and power constraints of mobile devices such as PDAs, cell phones etc. Also, there has been considerable work on improving the communication capabilities of mobile devices by increasing bandwidth and the development of protocols that are fault tolerant and resilient to the error prone wireless media. However, data management in such mobile networks will be key to the success of Ubiquitous Computing. In this report we will review some of the existing work in Data Management in Mobile Networks. In the final section, we propose a novel caching strategy for time varying data in Mobile Networks.

2 Data Management in Mobile Networks

In [6], the author elucidates the challenges in ubiquitous data management which prevent the direct application of traditional data management techniques in ubiquitous environments.The author then goes on to provide a brief overview of two ubiquitous data management projects viz Data Recharg-

ing and the Telegraph Project at Berkeley. The author provides a case for ubiquitous data management, in that, for computing to be ubiquitous in its true sense, there must be adequate and efficient techniques for the storage, retrieval and processing of data in pervasive environments.Traditional data management techniques would not be suitable in such environments given their dynamic topology, limited bandwidth and battery constraints.The various functionalities required by ubiquitous computing applications are

- Support for Mobility
- Context Awareness
- Support for Collaboration

Support for Mobility

Given that most of the users will be mobile in ubiquitous environments, there must be seamless transfer as users move from one service provider to the other.Mobile devices generally have wireless connectivity which is more error prone and less reliable. Therefore, to support data applications under intermittent connections, we need to have efficient and intelligent data caching strategies. One such technique would be to use the users' predicted future needs to prefetch required data when connected to the network. There is also a need to support location based queries. This requires efficient data structures for the storage and retrieval of location information as well as infrastructure for tracking mobile users. Location information can generally be obtained through GPS, which however, works only outdoors. Projects such as Cricket [8], RADAR at Microsoft Research [2] etc deal with location detection indoors.

Context Awareness

To exploit the potential of Ubiquitous Computing to the fullest extent, the devices need to be aware of the context within which they are operating.Context information could be obtained from multiple sensors or even from user input. In order to support context awareness, there is a need to support some understanding of the users context. Further we need efficient storage techniques for user knowledge, and inferencing there upon to provide valuable feedback to the users. Since most interactions will be real-time, the inferencing mechanisms need to be fast and be able to work with incomplete, and often conflicting data.

Support for Collaboration

One of the primary aims of Ubiquitous computing is to create communities of users to enable sharing of data and services. This requirement entails support for the “storage,maintenance, delivery and presentation of shared data”. When data is shared, maintaining synchronisation and consistency of shared data is an important requirement. We will need access control as to who or when a data item can be created, accessed or deleted. Traditional data management techniques wouldnt be completely suitable in this scenario,as we a need a more “flexible and light weight co-ordination function”. Given the dynamic topology and dynamic nature of collaborations, there will be a need to maintain a log or history of events/communications so that new members can join in and immediately sync up. Also, we may want to analyse the causes and outcomes of a collaboration. Thus, the system should be able to provide reliable storage for history data.

A common requirement imposed by the above functionalities is adaptivity, which is not handled by traditional data management techniques. Also, it is possible to relax rules regarding consistency and synchronisation, and allow the users to decide what is best suited for them.

The author further gives a brief overview of two projects on Ubiquitous Data Management.

Data Re-Charging: Profile Based Data Dissemination and Synchronization: In this project, the users predicted future data needs are used to cache data when the device is connected to the network. The users needs are expressed as profiles. " Profiles can be thought of as long-running queries that continually sift through the available data to find relevant items and determine their value to the user". Profiles of users' needs describe the types of data required by the user, priorities of different data items and user context obtained from various sources.

Adaptive Dataflow Query Processing: Since data flow is dynamic and incomplete in ubiquitous environments, traditional database query processing systems which are based on a static approach and optimised for batch processing will break down in these scenarios. These environments require real time, resilient and adaptive data stream processing engines that can gracefully adapt to changing data rates and user inputs. The Telegraph project [4] at UC Berkeley develops an adaptive dataflow processing engine which is dynamic and adapts to changing data rates, storage and communication requirements etc. Unlike traditional static query execution plans, Telegraph uses "eddies", which are dataflow control structures for routing data to query operators on a per item basis, for query execution.

In [1], the authors introduce Broadcast disks which can be viewed as multipe disks spinning at different speeds, broadcasting on the same channel, thereby augmenting the users memory hierarchy.The faster disks are higher in the memory hierarchy while the slower spinning disks occupy the lower levels.These disks are introduced in an asymmetric communication envrionment. The authors distinguish between two different causes for asymmetry, as capacity based symmetry caused by information flow, and physical asymmetry caused by device and bandwidth constraints. Such asymmetric communication environments generally employ the push model for data dissemination, where in high-end powerful servers broadcast data to the clients at regular intervals. Thus the broadcast channel acts like a part of the memory subsystem for the clients. The authors propose a technique in which multiple disks of different sizes and spinning at different speeds are superimposed on a single broadcast channel, thereby servicing requests for different data items at different

latencies depending on which disk the data is assigned to. Data items are assigned to different disks depending on their predicted future accesses. More frequently accessed data are assigned to faster disks while the low frequency data are assigned to slower disks. The number of disks, their relative frequency and size can potentially be optimised for certain clients provided their frequency and access pattern are known apriori.

The authors further bring out an interesting aspect: The interaction of the broadcast disks and users cache. Traditional cache management policies will not provide optimal performance in the presence of broadcast disks, since the rate of data availability is now governed by the relative speeds of the disks and the placement of data over the various disks. Cache replacement techniques can longer be based solely on the local cache access pattern but will also need to factor in the rates of data availability achievable through the broadcast disks.

Generation of the Broadcast Program: The broadcast program is generated by taking into account the access patterns of the various clients. An important issue to consider here is that, due to bandwidth constraints, optimising the broadcast program for one client decreases the performance for other clients. Therefore we will want to generate a broadcast program that is suitable for all the clients. Also, the authors point out that as the variance in inter arrival rate for a data item increases, so does the expected delay for that data item. The generated broadcast program must have the following properties

- Fixed inter-arrival rate for data items
- Well defined interval for periodic broadcast
- Utmost use of available bandwidth

The broadcast program generation in its simplicity has the following the steps

1. Order pages from most accessed to least accessed.
2. Partition the pages into different disks depending on access probabilities.

3. Choose relative frequency of broadcast for each disk. The relative frequencies must be integers.

4. Split each disk into smaller chunks of data which will eventually be the unit of broadcast.

5. Interleave the chunks from each disk to generate the broadcast program.

The authors then move on to discuss the interaction of the client cache with the broadcast program. Traditional client management techniques in pull based environments cache the most recently accessed data. However such techniques would not work well in the presence of broadcast disks since the broadcast program is generated based on the access probabilities of all clients and not a single client. However, if the broadcast program was tailored for a single client, then we could cache the most recently accessed data and move those data items to slower disks there upon. Therefore in the presence of broadcast disks, the cache replacement policy must depend not only upon the access probability distribution, but must also take into account the cost of acquiring a replaced page. "The optimal replacement strategy is one that replaces the cache-resident page having the lowest ratio between its probability of access (P) and its frequency of broadcast(X)."

Simulation studies were carried out to compare and evaluate the performance of broadcast disks and different broadcast programs for varying access probabilities and cache sizes. In general, broadcast disks outperformed flat broadcast when the access probabilities were non-uniform. Also, the performance of a caching policy that approximates the ideal cost-based cache policy is compared with LRU. This policy is shown to provide better performance in the broadcast disk environment than the traditional LRU policy.

An important contribution of this work has been the use of multiple disks at different speeds superimposed on the same channel thus providing different data at different frequencies. Also the interactions of the broadcast program and the user client cache management strategies are noteworthy. The authors in this paper assume that only the server modifies data. It would be interesting to investigate the effect of client modifications on the broadcast program and cache management.

In [3], the authors propose a novel framework called Relay Peer Based Caching Consistency(RPCC) for maintaining consistency of data in Mobile Ad Hoc Networks(MANETS). They develop a hybrid approach for co-operative caching in MANETs where in both push and pull are used for cache invalidation. In traditional mechanisms, an invalidation report is broadcast by the source at regular intervals of time. However, the authors claim that such a mechanism will be useful only in stable networks and will fail in MANETs, given their dynamic topology, limited bandwidth, energy constraints and peer-to-peer model of communication. Therefore the concept of a relay peer is introduced. Typically a relay peer node is one with high sharing capability and low relative mobility, and is within a defined vicinity of the source. The source pushes data to relay nodes while the other cache hosts pull the data from relays, thus achieving the benefits of both pull based model(lower latency) and push based model(lower network bandwidth). Relay peers are selected among those nodes which are within the TTL limit of the invalidation message broadcast by the source, based on the following three parameters

- CAR-Coefficient of peer access rate
- CS-Coefficient of Stability
- CE-Coefficient of Energy

which are calculated every period of time ϕ.

Host Access Rate(HAR) is defined as the number of cache accesses(N_a) at a host during the time ϕ. Therefore $HAR = N_a/\phi$.To account for the history of cache accesses, HAR at time t, HAR_t is defined as

$$HAR_t = HAR_{t-2} * \omega/4 + HAR_{t-1} * \omega/2 + N_a/\phi * (1 - \omega/4 - \omega/2).$$

where ω is the weighting factor for the more recent values.

Also, CAR at time t, is now defined as

$$CAR_t = 1/(1 + HAR_t)$$

Similarly,Host Switching Rate(HSR) and Host Mobility Rate(HMR) are calculated as,

$$HSR_t = HSR_{t-n} * \omega + N_s/\phi * (1 - \omega)$$

$$HMR_t = HMR_{t-n} * \omega + N_m/\phi * (1 - \omega)$$

where N_s is the number of times a node switches between connected and disconnected in time ϕ, and N_m, the number of times the node moves from one subnet to another. CS is now defined as

$$CS = 1/(1 + HSR_t + HMR_t)$$

HER_t is defined to be the peer energy level at time t, and E_{MAX}, the maximum energy level. CE is now defined as,

$$CE = HER_t/E_{MAX}$$

Thresholds are defined for each of the three parameters and a node which satisfies all the three threshold constraints and is within the TTL of the invalidation message can be selected as a relay node.

The Protocol: The source periodically broadcasts an Invalidation message. On receiving the Invalidation, a node satisfying the threshold constraints can apply to the source to become a relay peer. The source then sends an ACK and adds the node to the set of relay peers. A relay node, on the other hand, on receiving the invalidation message checks the status of its cached copy to ensure its copy is upto date. If the relay node has outdated data, then it polls the source for the latest version and, the source responds with the updated data. Also, if a data item has been changed at the source, the source sends out an UPDATE message to the relay nodes, asking them to update their cached copies. If a cache host cannot determine if it has the latest version, it polls the relay nodes. The relay nodes respond either immediately, or after receiving an invalidation message from the source, informing the requesting node that it has the latest version or asking it to update.

The RPCC model can support various levels of consistency such as strong consistency, δ consistency and weak consistency by tuning appropriate parameters in the model. Also, the RPCC

model is resilient to node disconnection since a re-connected node can query the Relay peers for the up-to-date version of the data. Simulation resutls show that the RPCC model required lower network bandwidth and had lower latency for cache requests compared to the pull and push models.

This work is interesting in that it creates an overlay of relay nodes for cache invalidation in MANETS. It would be interesting to see how the number of relay nodes required to ensure a specified performance level, varies with the number of nodes in the network and different mobility models. However, it is not very clear how N_m, the number of times a node moves from one subnet to another, is calculated in an infrastructure less MANET.

In [9], the authors propose an adaptive content based routing mechanism for routing search queries in mobile networks. Each node constructs a synopses of its data, and disseminates the synopses to the nodes which are most likely to benefit from the synopses. The content synopses are then used to construct a data driven routing mechanism where in a search query is routed to the node which is most probable to serve the search request.

Given the dynamic topology and energy constraints of mobile networks, we would like data to reach the destination, keeping the number of propogated messages minimum. It is in these cases that content based routing proves to be useful. Each node uses Bloom filters to construct a synopses of its local data and disseminates it to its peers which are then used in the routing mechanism. The authors discuss three strategies for synopses dissemination

- Immediate Local(IL):In this strategy, the synopses of a node is broadcast to its immediate peers. Although, this generates less traffic compared to the other strategies, it is not useful in content based routing since there are only a limited number of synopses available for making routing decisions. This leads to the node randomly querying a peer, which in general generates more traffic.

- Adaptive Local Remote (ALR): In this strategy, the synopses of a node are sent not only to its immediate peers, but also to other peers in the network. These peers are selected among those that would benefit most from the synopses. As a result the content synopses is

sent to those peers which have queried the node frequently and, nodes which have received responses from this node.

- Adaptive Local Remote(ALR): In this strategy, not only the local synopses but also the stored synopses of remote peers are exchanged. Thus the nodes expand their horizon of availability in the network. Although this scheme generates comparatively more traffic, more intelligent routing decisions can be made as there are a lot more synopses to look at.

The scheme handles content updates by grouping and sending Bloom filter updates when there is a content change. The authors suggest different dissemination strategies for static and dynamic peers. In the case of dynamic peers, the node solicits the content synopses of its immediate neighbours for routing decisions, while on the other hand, the nodes push their content synopses to static peers so that the information required for routing is already available with static peers. Node disconnection is handled by removing the node's content synopses after a certain period of detecting disconnection.

Simulation results indicate that content based routing outperforms flooding based routing in terms of bandwidth and power usage. In Content based routing, the ALR scheme seems to have the least query latency, acceptable overhead and high accuracy as it has more synopses available to make intelligent routing decisions.

However, one of the interesting issues is how do we distiguish between a static peer and mobile peer. This distinction is subtle since the authors suggest using different content dissemination strategies for these classes of nodes. Particulary in the ALR scheme which caches synopses of remote peers,false prediction of a node's mobility could lead to severe performance degradation as there would be the overhead of transmitting the synopses as well as the penalty of incorrect routing.

3 Caching Time Varying Data in Mobile Networks

In this work, we attempt to address the problem of caching time critical data in ad hoc networks. Particularly we are interested in the caching strategies for the mobile terminals. Normally, cache invalidation is based on a time out mechanism or invalidation Report. However, these mechanisms do not consider the distance between the cached copy and the data source. We attempt to factor in the distance between the cached copy and the data source in the aging process for cached data. Therefore, nodes which are closer to the source can cache a data item longer than those far off. We use the hop count as an estimate for the distance. The hop count between a cached copy and the data source can be obtained from the routing protocol.

Our motivation is based on applications that need to work with rapidly changing data e.g. Stock Market feeds, etc. In these scenarios, when the source modifies data and sends out a cache invalidation message, the invalidation message is more likely to reach a far off node much later after the node has actually read the stale data. On the other hand, if the caching node times out these far off data quickly, it does not run the risk of using stale data. Also, nodes which are near to the source, will receive cache invalidation message soon enough, that they can cache data for a longer time than those far off without the risk of using invalid data. We use a weighted sum for the data aging process

E.g. Consider the function

$$\text{Data\_age}=\alpha * \text{hop\_count} + (1- \alpha)\ \text{time\_in\_cache},\ 0 \leq \alpha \leq 1$$

Here α determines the importance of hop count in the aging process.

For e.g. for α =0.5 and cache time out =2

We see that for hop count =1, time_in_cache can be upto 3 before the data is timed out, while on the other hand for hop count =2, time_in_cache can be only at most 2 before time out.

Thus larger the hop count, smaller the time_in_cache for a given cache time out.

We also investigate the issue of tagging confidence levels with cached data. Under this scheme, when a cache satisfies a data request, it also associates a confidence level % such as 90% to the data

it supplies. The source always supplies data with 100% confidence level. Different applications on the nodes can now request data of different confidence levels and the first cache which satisfies the required confidence level will satisfy the request. We can use the Data_age computation from above in computing the confidence level of a cached data item.

Also our scheme can fit into existing co operative caching schemes. For e.g. If the neighboring nodes overhear the cache request interaction and determine that the confidence level of their cached copy is lower than the reply, then they also cache the data with the higher confidence level. Also the confidence level associated with the cached data will need to reflect the confidence level of the data source.

Much of our work is still in preliminary stages and we hope to address the following issues

- Estimation of data transmission time along a route
- Confidence level computation
- Development of an analytical model
- Comparative analysis with existing schemes
- Development of simulations that test the model

References

[1] Swarup Acharya, Rafael Alonso, Michael Franklin, and Stanley Zdonik. Broadcast disks: data management for asymmetric communication environments. pages 199–210, 1995.

[2] Paramvir Bahl and Venkata N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *INFOCOM (2)*, pages 775–784, 2000.

[3] Jiannong Cao, Yang Zhang, Li Xie, and Guohong Cao. Consistency of cooperative caching in mobile peer-to-peer systems over manet. In *ICDCSW '05: Proceedings of the Third Inter-*

national Workshop on Mobile Distributed Computing (MDC) (ICDCSW'05), pages 573–579, Washington, DC, USA, 2005. IEEE Computer Society.

[4] S. Chandrasekaran. Telegraphcq: Continuous dataflow processing for an uncertain world, 2003.

[5] Mitch Cherniack, Michael J. Franklin, and Stan Zdonik. Profile-driven data management.

[6] Michael J. Franklin. Challenges in ubiquitous data management. *Lecture Notes in Computer Science*, 2000, 2001.

[7] Filip Perich, Anupam Joshi, and Rada Chirkova. *Data Management for Mobile Ad-Hoc Networks*. Springer, July 2005.

[8] Nissanka Priyantha, Anit Chakraborty, and Hari Balakrishnan. The cricket location-support system.

[9] Thomas Repantis and Vana Kalogeraki. Data dissemination in mobile peer-to-peer networks. In *MDM '05: Proceedings of the 6th international conference on Mobile data management*, pages 211–219, New York, NY, USA, 2005. ACM Press.

[10] Chara Skouteli, George Samaras, and Evaggelia Pitoura. Concept-based discovery of mobile services. In *Mobile Data Management*, pages 257–261, 2005.