Mobile awareness in a wide area wireless network of info–stations

Tao Ye
Computer Science Devision,
UC Berkeley
tye@cs.berkeley.edu

H.–Arno Jacobsen
ICSI 1947 Center Street, Suit 600
Berkeley, CA 94704
jacobsen@icsi.berkeley.edu

Randy Katz
Computer Science Devision
UC Berkeley
randy@cs.berkeley.edu

Abstract

Wireless networking is becoming an increasingly important communication means, yet high wide–area wireless data connectivity is difficult to achieve due to technological and physical limitations. To alleviate these problems we experiment with an alternative by placing many high bandwidth local "islands" of *info–stations* dispersed throughout the low bandwidth wide–area wireless network. The location and distribution of the individual stations is crucial for the network's overall effectiveness, as demonstrated by our investigations. The info–stations are deployed in a transparent manner, often not geographically visible to the user. Applications must be designed to be mobile–aware and able to account for changing network characteristics by optimally utilizing the available network resources. We simulate alternative network layouts and determine their effectiveness by experimenting with an incremental map downloading application for road travelers that uses intelligent prefetching to take advantage of the info–stations. The prefetching algorithm uses location, route, and speed information to predict future data access. Our experiments show substantial performance improvement of the mobile–aware application in the info–station network over a mobile–unaware application in a conventional wide–area wireless network. The prefetching algorithm proves to hide latency from the user better than a naive prefetching algorithm. We have achieved between 16% to 50% improvements, depending on the prefetched amount. The results also suggest that a network design with frequent short range info–stations is better than one with fewer, longer range stations.

1 Introduction

The past few years have witnessed the rise of a new computing model — distributed mobile computing. Combined with a growing wide area wireless communication infrastructure, whole new applications have become feasible, such as, navigational aids on the road, mobile offices in the car, wireless classrooms, and personalized information services, to just name a few. A paradigm of anytime–anywhere–computing has largely become possible as the result of the combination of these two technologies.

Today's wide area wireless networks, as the key component of the wireless communication infrastructure, still suffer from low bandwidth and frequent disconnections. Covering wide areas with high bandwidth requires complex equalization, due to signal attenuation, multipath fade, and shadowing effects. Currently, both CDPD and GSM networks offer merely a transfer rate of 5–10kbps. The microcell network, Ricochet [Met95], by Metricom Inc., provides up to 30kbps bandwidth, but it does not support access from moving vehicles.

Emerging, third generation networks, investigated in Europe under the umbrella term UMTS (Universal Mobile Telephone Services) aim at supporting up to 2 Mbps services on mobile links [XII96, KLO97, BGR$^+$98, O'M98]. Mobile services for the use in and the control of high–speed trains are currently pursued [ICG$^+$97]. Mobile broadband systems (MBS), [CP93, Zub94, JP94], even aim at supporting data rates of up to 155 Mbps, for quasi-mobile users within the range of stadiums, factories, or buildings. HIPERLAN (High Performance Local Area Network) [Kru93, Hal95] is a suite of standards defined by the ETSI that supports data rates of up to 20 Mbps, later aiming at 155 Mbps (HIPERLAN 4), for local environments (50 m) with limited mobility.

Sophisticated radio engineering will lead to improved bandwidth, coverage, and mobile access, but this will lag the wire–line environments, in terms of both capabilities and cost. Compared with the wired network, the bandwidth available in a wireless link will always be several orders of magnitude lower. We believe, however, that higher performance can be achieved by combining intelligent network layout with efficient applications design.

One idea is to lay out the wireless network such that high bandwidth "information islands" lie in a sea of relatively low bandwidth coverage [TMOIT94b, TMOIT94a, FI96]. Such a network configuration provides high bandwidth, but not uniformly. Combined with network–aware applications this benefits a large class of non–latency sensitive applications, including interactive map querying, e–mail retrieval, and web browsing [FI96, Bad97].

Badrinath, [Bad97], was among the first to propose infrastructure for supplying information services, such as, e–mail, fax, and web access by placing 'information kiosks' at traffic lights and airport entrances. He assumes that users will move to kiosks for information access [Bad97].

By constructing info–stations along major highways we aim to support highly mobile users, such as drivers, with non–real–time media services. We emphasize hiding user perceived latency through network aware application design, as demonstrated initially in [YJ97]. Thus, the major differences between our approach and the ideas sketched

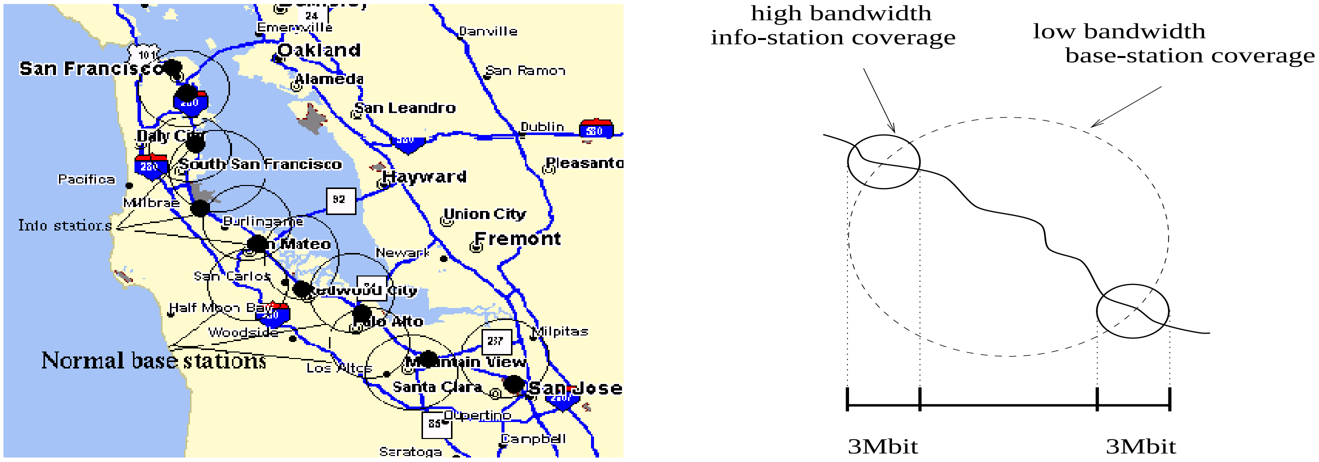


Figure 1: The map illustrates a possible layout of info–stations. The black dots denote the areas covered by info–stations. The circle around them denote the cells of the regular low bandwidth wide–area wireless network. (This map is only an illustration, it doesn't corresponds to any real cellular network layout.) The right most figure gives a conceptual account of the relation between base and info–station.

in [Bad97] are our objective to support highly mobile users and our focus on devising and evaluating techniques for building network–aware applications.

Most of today's applications are not designed to be network aware. For the class of applications we are targeting, network awareness establishes a connection between the changing network availability of info–stations, the user interactions, and the application design. Some work on event signaling [BW96], already supports this kind of mobile awareness. Watson, [Wat94] summarizes several design techniques helpful for mobile application design. In this paper we investigate the effectiveness of info–station networking combined with mobile–aware application design and study their synergetic effects in a simulation.

We focus on a network–aware client–server application: Map–on–the–Move, an incremental map retrieval system, supporting vehicle drivers. Map–on–the–Move automatically refines maps in response to user movements. This application is simulated by a workload model that employs intelligent prefetching, a technique we developed to make the application mobile–aware. Prefetching decisions are made according to the client's geographic location, driving direction, speed, and the proximity of info–stations. Moreover, we are interested in efficiently laying out the islands of info–stations, given their communication characteristics and information about the traffic load of the region covered. The Map–on–the–Move model is used to drive the study of the optimal configuration of info–station networks. Figure 1 shows an example of a possible info–station layout for the San Francisco Bay Area and illustrates the relationship between info–station and base–station coverage.

The remainder of the paper describes and experimentally validates our approach. Section 2 describes the architecture of the Map–on–the–Move application and discusses the intelligent prefetching scheme. Section 3 presents the model we have developed to study the approach. Section 4 experimentally evaluates the techniques proposed. We have deferred a discussion of related work to Section 5, to be able to better place our work within the context of existing approaches. We discuss future work and draw conclusions in Sections 6 and 7, respectively.

2 Application design for the network of info–stations

Network–aware applications place special demands on the wireless networking environment and the application design. In this section, we specify Map–on–the–Move in detail and extract its requirements. We proceed by designing a client–server architecture implementing Map–on–the–Move in a network–aware manner. Finally, we describe the intelligent prefetching algorithm that ties the application more closely together with the network layout.

2.1 Map–on–the–Move

The objective of Map–on–the–Move is to deliver maps, at the appropriate level of detail, on demand for instantaneous route planning, emergency use by police and firemen, and for the use by various groups of field workers. The application can be interpreted as a 'mobilized' version of an Internet map service, like Mapquest [Map96]. We now motivate its deployment and operation.

At the beginning of a trip, the user inputs source and destination location. Map–on–the–Move retrieves the corresponding top level map that highlights the route. As the trip unfolds, more map segments will be automatically fetched into the mobile client's cache. The application decides autonomously, given the driving route, the vehicles position, driving speed, and direction, which information to retrieve. The user will periodically demand more information about the surrounding area, either looking for more detailed information about the current location, or seeking more information about the route further ahead, i.e., nearby restaurants, traffic updates, road conditions, tourist information et cetera. In both cases the access pattern will be closely related to the current vehicle location, speed, and direction of travel, as well as the route plan.

For example, if the user is driving 65 mph on a highway, she is not likely to demand detailed map segments. On the other hand, if she is driving 20 mph on local roads, she will most likely be looking for more detailed information. An intelligent prefetching algorithm will access the right information, at the right time, and correct level of detail, to have it ready in the cache for the user, thereby improving latency and saving bandwidth. The algorithm can also take advantage of the topology of the network of info–stations by increasing its activity in high bandwidth regions.

A more detailed discussion will follow in Section 2.3. The following section describes the system architecture.

2.2 System architecture and design

2.2.1 Design goals and rational

The Map–on–the–Move mobile application design is governed by the following design goals:

1. The need for a simple and cost effective solution to increase data availability in mobile and wireless environments;

2. Support thin clients in mobile environments;

3. Hide network latency from the user by making the application mobile–aware;

4. Experiment with just–in–time delivery of dynamic data;

5. Experiment with the info–station capability and mobile–awareness supporting it.

The overall design follows the client–server paradigm with a request passing from the client through the info/base–station to the server. Figure 2 depicts the individual stages in more detail. The approach is purely pull–based. We discuss possible alternatives in Section 7.

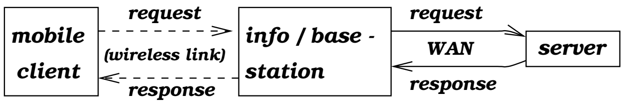


Figure 2: Request–response pattern of client–info /base–station–sever interaction.

One of the primary design goals is to keep the overall system simple and cost effective. We chose incremental retrieval of the map instead of downloading the complete detailed map, because the latter is expensive and inefficient, as the user might not need all the details. For example, fully rendered San Francisco bay area map is estimated to be in the 20MByte range, but a detailed segment that's of interest to the driver, as used in the simulation, is only 20KBytes. The goal is further stressed by the desire to support thin clients. Devices installed in vehicles or carried by the users tend to have limited display functionality and computing power. Hence, more functionality needs to go in the server and network infrastructure. In particular, we trade off computing power with bandwidth by transmitting the fully rendered map which is large in data size but requires only simple display capability. To hide latency and improve availability of the data at the client site, we design the application to be aware of the high bandwidth info–station links and use intelligent prefetching techniques.

Since the data transferred to the client is large, a dissemination based approach [AFZ96] would not be feasible. Using permanent storage, such as CD-ROM or hard disk, is also nonoptimal because, (1) thin clients are desirable; (2) road networks frequently undergo changes; (3) data on CD-ROM is not updated easily, therefore time dependent information related to location, such as traffic conditions, weather updates, changing entertainment and tourist information cannot be included.

Our approach generalizes to related applications, like web–browsing, where depending on the document large bodies of graphic or sound data may be transferred.[1]

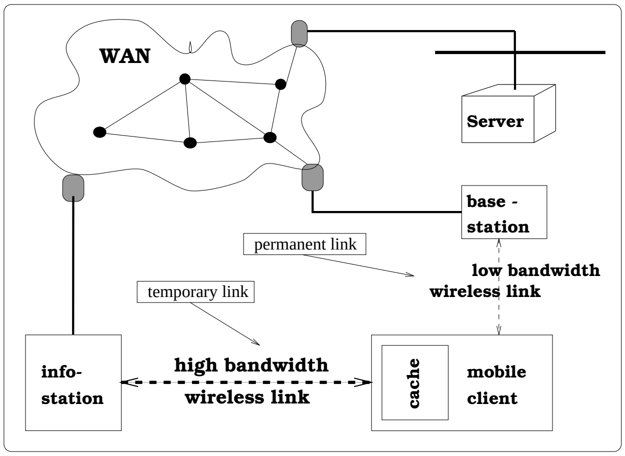


Figure 3: Map–on–the–Move architecture and client interaction with info–station and normal base–station.

2.2.2 Architectural layout of the network of info–stations

Figure 3 shows the conceptual architecture of the network of info–stations. A mobile client is connected to a wireless network, which in turn connects to a map database server on the Internet. Within this wireless network there are two types of links, pockets of high bandwidth near info–stations and low bandwidth in between. The mobile client uses high bandwidth links when it is within info–station coverage. Outside these regions, client requests are passed to the server via a conventional cellular base–station. The wireless links will in general be much slower then the wired connection from the info/base–station to the server.

Whenever a mobile client enters the coverage of an info–station, the associated high bandwidth communication characteristics become available, unless some other intersecting region offers superior characteristics. For now we have neglected this case and assume that at any point in time only a single info–station can be reached by a client.

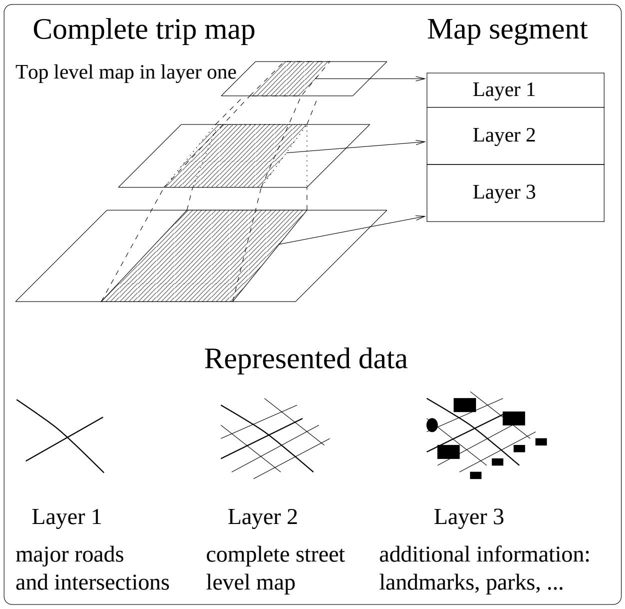
2.2.3 Map Representation

We assume that the map data sent across the network to the mobile client is in a fully rendered but layered format, and design the map server accordingly. The map is partitioned into map segments of equal data size. Each map segment also represents a two–dimensional equally sized geographic region of the original map. However, map segments are designed such that they reveal access to successively increasing levels of details in the original map. Each segment contains

[1] Prefetching decisions could then, in the simplest form, be based on sub-links in documents.

three blocks which represents three increasing layers of detail. These are:

1. `Layer 1`: represents major roads and intersections,
2. `Layer 2`: represents a complete street–level map of the area covered,
3. `Layer 3`: represents additional detailed information, like landmarks, parks, tourist information, et cetera.



Complete trip map
Map segment
Top level map in layer one
Layer 1
Layer 2
Layer 3
Represented data
Layer 1
Layer 2
Layer 3
major roads and intersections
complete street level map
additional information: landmarks, parks, ...

Figure 4: Conceptual representation of map and partitioning into map segments.

Layer 1 corresponds to the highest level map (i.e., the least detail) and needs the least storage space. Layer 2 and 3 reveal incrementally more detail. Their storage requirements are much larger than that of layer 1. The different map segments may be retrieved independently but need to be loaded incrementally to obtain the full information stored in the map. Figure 4 illustrates this representation in more detail. Each map segment is stored by a certain number of pages. The exact storage requirements for the individual layers are given in Section 3.

A client request to this map contains a map segment number and the desired detail level l. The response will be all the data up to layer l in that segment. A user can request the map of a region that consists of several map segments. We simply use several client requests together. The request–response operation is implemented on top of a TCP connection similar to common RPC implementations. A single TCP connection is established for each request.

2.3 The intelligent prefetching algorithm

To alleviate user perceived latency at the mobile client, we enhance the demand–page–caching cycle by prefetching pages which are likely candidates for future accesses. Our prefetching algorithm has two main characteristics. First, we use the user's location, speed and driving direction, combined with route information to select data for prefetching. Second, to add mobile–awareness to our application, prefetching is enabled when the user comes within range of an info–station.

Most prefetching techniques suffer from the lack of information indicating future application needs. However, for the class of location dependent applications we study, much information about future needs can be derived, especially, due to their location dependent nature and well understood semantics of the highway and street network.

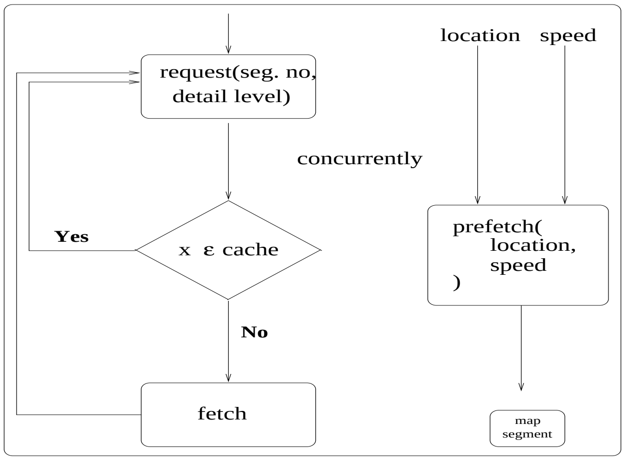


Figure 5: Flow chart of the simple cache–prefetch algorithm.

We assume that the user has given the start and end points of the trip. This allows the application to considerably narrow down the set of pages potentially accessed by the user. Secondly, current location, driving speed and direction are used as indicators to predict future paging needs. This heuristic is based on our assumptions about the user request pattern detailed in Section 3.2. For example, we assume that a client moving rapidly is less likely to access detailed information about the area around and ahead of her as one who is traveling more slowly.

The basic prefetching loop run by the mobile client is outlined in Figure 5. Whenever the mobile client enters the coverage of an info–station, it prefetches a fixed amount of k bytes. For clients with different driving speed and location, the amount k represents different maps. When a client is moving along at a fast speed, the amount k represents less levels of detail but more number of segment in a map (i.e., a larger region). A slower moving client will receive this k bytes as more levels of detail but from fewer segments (i.e., a smaller region).

We discuss the experiment of choosing the right prefetching amount in Section 4. By relating prefetching to the info–station layout, we periodically take advantage of the available high bandwidth. In later sections we demonstrate the benefits of such application–network coupling over an application that is unaware of the network bandwidth variation. It is possible that such coupling could happen in the network layer, as suggested in [Bad96], but we leave that as a future network layer design option. Mechanisms for notifying the application of network variation, such as high bandwidth availability, are discussed in [BW96].

Map server model parameters	
`SrvDBSize`	varies from trip to trip
segment detail level and corresponding size	
layer 1	1 Kbyte
layer 2	10 Kbyte
layer 3	10 Kbyte

Table 1: Server model parameters. The corresponding size of a segment detail level represents the amount of data returned from the server for a request with the specified detail level.

Client model parameters	
Device parameters	
cache size	0-4 MByte
replacement policy	FIFO
User request pattern generation	
TripTime	30 min – 1h 30 min
ThinkTime	{`local, highway,` `combined` }–pattern
Workload: Zipf–distribution	
skew	ρ
translation	k

Table 2: Client parameters. (See text for explanation.)

3 Modeling the mobile environment

We simulate the Map–on–the–Move application and the network of info–stations. This section motivates and develops the models we used. The model of the mobile environment contains the network of info–stations, a mobile client, and a server.

The info–station network model specifies the network link characteristics between the client and the server. This model accounts for the different communication characteristics of that link, according to whether communication is via an info– or a base–station. The client model specifies the mobile client's parameters and the user request patterns. The server model models the map database server.

For the purpose of this study we assume that all of the traffic is generated by our application. Furthermore, we assume that all client requests can be served either by an info–station, if the client is within its coverage, or by a regular base–station, otherwise. Congestion would impact the availability of data at the client and therefore influence its performance. However, this can be handled at a lower network layer by mechanisms like fair queuing. Alternatively, a rate limiting scheme for clients could be introduced. We therefore neglect background traffic and possible congestion in the simulation.

Next we describe the server, the client, and the info–station network model in more detail. We then show how we derived the model parameters by explicitly surveying state–of–the–art wireless communication infrastructures and their technical specifications.

3.1 The map server model

The map server model constitutes a fully rendered trip map, stored as individually retrievable map segments. A segment of the size specified in Table refserver roughly corresponds to a 1 mile long, 5-10 miles wide region. A query is performed based on (x,y)–location–coordinates and a desired detail level. The map server model returns the appropriate amount of data for a specified detail levels. The representation of maps and the query model has already been discussed in Section 2.2.3.

The map server model is characterized by its size and content. The size is specified by a single parameter, `SrvDBSize`, representing the number of map segments for one trip. The `SrvDBSize` varies for each trip taken. Table 1 summarizes these values.

3.2 The mobile client model

The mobile client model represents a user requesting map information while driving. The model defines the mobile client device, specifying its cache size, cache replacement policy and the user request pattern. The user request pattern is determined by the `TripTime`, the actual request pattern chosen and the workload distribution. The model parameters are given in Table 2.

The `TripTime` parameter models the length of a trip, i.e., the time the client is operational requesting information from the info–station network.

The `ThinkTime` parameter models the time between requests generated by the client. It accounts for the user's interaction behavior with the system. We use it to model several alternate user behaviors. The following request patterns are represented:

- *local–request–pattern*, modeling a driver in a downtown area;
- *highway–request–pattern*, modeling a driver on a highway;
- *combined–request–pattern*, modeling a trip on a highway between two downtown areas.

The *local–pattern* models a user who generates a few (i.e., between three and six) requests with inter–request times distributed uniformly within the range of 10 to 30 seconds and another few requests after 5 to 10 minutes. This pattern is repeated until the overall `TripTime` exceeded. It models the situations of driving in a city, i.e., searching, driving, searching, driving in a periodic cycle.

The *highway–pattern* is similar to the local–pattern, except that the gap between repeated requests is larger. We assume that as a driver runs down a highway, she will less often interact with the map retrieval device. The gaps are therefore uniformly drawn from the interval of 10 to 20 minutes.

A *combined–pattern* is generated as a hybrid of the above two. It models the situation of looking for the highway entry, driving to the desired exit and finally finding the way locally to the destination.

The workload, from which individual page requests are generated, models a driver demanding map segments from the user–interface of the map loading and displaying device. We assume that the distribution of these requests is localized around the area of the user's location, with a decaying probability of access for map segments further towards the trip destination. We assume that a user gets lost occasionally, i.e., once per each 30 minute trip and needs to backtrack. The model accounts for this by requesting pages that is around a location slightly behind the current position in

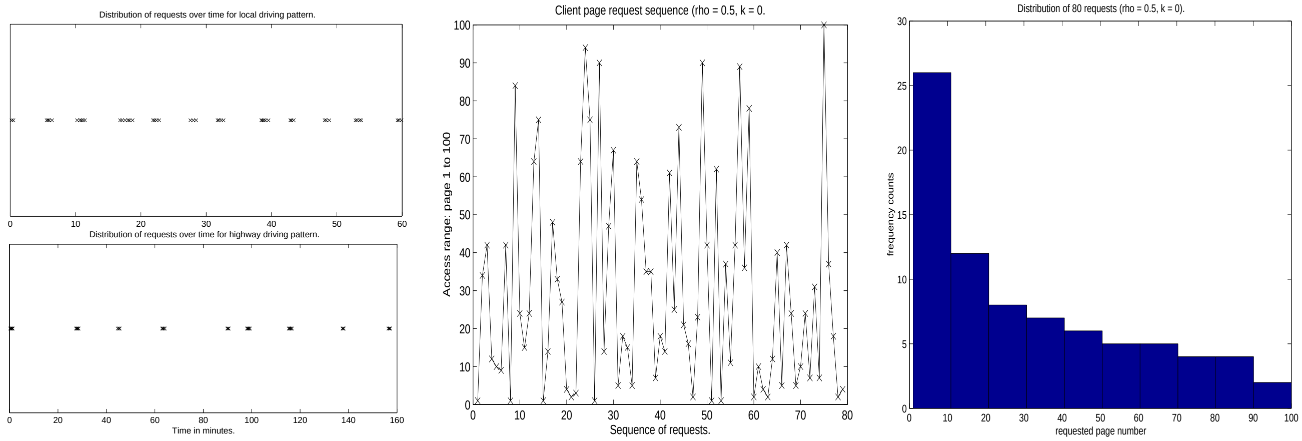


Figure 6: The left most graph actually contains two graphs, depicting the request pattern of the client over time. The upper one shows the *local request pattern*, i.e., a repeated stop–and–search cycle. The lower one shows the *highway request pattern* which models a more prolonged stop–and–search cycle, with reduced periodic requests than the local pattern. The middle graph plots the requested page numbers over time which have been drawn according to the generalized Zipf–distribution with parameters ($\rho = 0.5$ and $k = 0$). The right most graph depicts the frequency count of these requests. We assume the page request naturally follows the Zipf–distribution.

the trip. Page requests are drawn randomly from a range of m to n, where n is the total number of pages stored in the data base (i.e., `SrvDBSize`) and m a dynamically varying value derived from the location and speed measurements of the client's location. Page requests are generated according to a generalized, two–parametric Zipf distribution. These distributions are often used to model skewed local access behavior (e.g., [FZ96] and [Dag88]). Figure 6 shows a typical sample drawn according to this law. The distributions are defined by:

$$f_X(x) = B(x+k)^{\rho+1}(x = 1, 2, ...),$$

where $\rho > 0, k \leq 0$, and B is a constant. For k integral, this is a truncated one–parameter distribution, translated by k units. For $k > 0$ the distribution has relatively less probability concentrated at $x = 1$ (compared to $k = 0$). The parameter ρ controls the skew of the distribution. For increasing ρ the concentration of the distribution on one increases.

3.3 The info–station network model

As outlined above a mobile client moves through "islands" of high bandwidth followed by extended periods of low bandwidth. The high bandwidth islands correspond to regions closely surrounding the info–stations. Stretches of low bandwidth correspond to regions where communication to the server is via a base–station.

The info–station network model accounts for these dynamically varying characteristics of the client–server communication link. It represents the link by its bandwidth and delay parameters.

The distribution of info–stations in the network is described by two parameters, `density` and `coverage`. `Density` describes the percentage of the overall area covered by info—stations, while `coverage` describes the physical range of one info–station. Both parameters should be described in two dimensional space. Here, for simplicity, we simulate `coverage` in a one dimensional model, as the length of time a mobile client stays in an info–station coverage, relative to a constant driving speed. `Density` is calculated as the percentage of time spent in info–station coverages over the entire trip time. Table 3 summarizes the parameters.

Info–station network model parameters	
info–station parameters	
bandwidth	3 Mbit
delay	50 msec
coverage	1-2 min. client time
density	0-60%
base–station parameters	
bandwidth	10kbps
delay	300msec

Table 3: Info–station network model parameters.

3.4 Mobile environments and their real–world characteristics

This section surveys state–of–the–art wireless networks and their characteristics. The result is presented in Table 4. The information has been obtained from the numerous infrastructure providers' web pages. The providers are referenced in the table. The network parameters in the experimental study have been chosen to reflect these real world characteristics.

4 Experiments and Results

This section presents our experimental methodology, measurements, and summarizes the results of our study.

Description	Bandwidth	Fixed Cost Estimate	Variable Cost Estimate	Coverage	Comment
Data Pager	500, 1200, 2400 bps	$200	minimal	Throughout US	
Cell Modem	14.4-230kbps (4.8–14.4 kbps typical)	$270–$390+ cell phone	monthly fee .30 $ minute	Throughout US	
Metricom	30kbps	$ 300	$ 30 monthly fee only	Bay Area, Washington D.C., Seattle, LA	Not for vehicular speed
CDPD	19.2kbps (9600bps typical)	$1000	monthly fee+ $ .10/KB	Major Metropolitan areas	Scattered throughout US
IBM ARDIS	19.2kbps	$800–1000	monthly fee + $.30–.59/KB	Major Metropolitan areas	
RMD	8kbps (2400-4800bps typi.)	$750	monthly fee	Major metropolitan area	long latency
GSM	9.6 kbps	$20–$100	monthly fee + amount per Kbyte	world wide	
UMTS	144 kbps - 2 Mbps	experimental		aim: Europe wide	vehicular speed
MBS	155 Mbps	experimental		stadions, factories (about 1 km)	mobility 50 km/hr
HIPERLAN	20 Mbps	vendor specific		50 m	35 km/hr

Table 4: Summary of wireless network provider options.

4.1 Experimental methodology and experimental setup

All experiments are performed with the network simulator *ns* [Mcc], which we extended with a client–server model incorporating different caching and prefetching strategies. The primary performance metric used, to monitor the application, is the response time as perceived by a user. Each experiment was repeated five times for a given set of parameter values. The response time is measured for each user request and averaged over the experimental runs. One experimental run corresponds to a single trip. We monitor the variance of the individual runs by also computing their standard deviation. Measurements are all performed with warm caches, i.e., we start measuring after the application had been running for some time. This is done to not distort our performance metric with initial cache loading effects. The response time of one request is determined as follows:

1. Check if the requested pages are in the cache. If yes, set response time as a fixed cache access overhead and go to (3), otherwise continue. The cache access overhead is set to 0.001 msec. Cache size varies from 0-4MByte in different experiments. The results of that variation is discussed in Section 4.2.

2. Determine the size of the requested transfer, record the start time, establish a TCP connection and retrieve the data through the info–station or base–station link, depending on the current connectivity. (This is done transparent to the user.) After the retrieval is completed, record the finish time and calculate the response time.

   In the link model we ignore the delay from the info–station to the server for several reasons: (1) We are primarily interested in the behavior of the client with respect to the changing characteristics of the wireless link; (2) the delay on the wire network is a different study.

3. Record the response time.

A secondary performance metric used to monitor the application is the hit–rate on the prefetched data, which is an indicator for the overall effectiveness of the intelligent prefetching technique. The number of pages to prefetch for one request is evaluated against the cache and possibly reduced by the pages already there.

Each experiment models a trip taken according to different driving scenarios, derived from the user–request–patterns presented in Section 3.2. Two main scenarios are constructed with variable mobile client and info–station parameters:

- *Local driving scenario.* This scenario models the situation of a client driving in a downtown area where the chance of encountering an info–station is relatively high. The `SrvDBSize` parameter is set to 100, a small value, representing a small number of available map segments. The `TripTime` parameter is set to 30 min. The '*local–request –pattern*', as described in Section 3, is used as the underlying user model.

- *Highway driving scenario.* This scenario models the situation of a longer trip, i.e., between 1 and 1.5 hour long.

  The user goes through three phases in such a trip — (1) from the starting point of the trip to the highway entrance; (2) drive on highway; and (3) from highway exit to the destination of the trip. The user request pattern alternates accordingly, from '*local–request–pattern*' to '*highway–request–pattern*' and back to '*local–request–pattern*'. The `SrvDBSize` parameter is set to 300, representing a larger number of available map segments. The `TripTime` parameter is set to 90 min.

In both scenarios we introduce occasional referencing of map segment previously touched. We add a few such requests every 30 minutes. This is to model when the driver makes a mistake and decides to backtrack.

4.2 Experimental goals and results

The experiments are designed to study the effectiveness of the network of info–stations as compared to a traditional wide–area wireless network. Furthermore, we explore the effectiveness of alternative info–station layouts under different driving scenarios. This is to find design guidelines for areas with different request densities, i.e., downtown versus highway. We also study the behavior of mobile–aware applications in such environments for different design parameters. In particular, we focus on finding the optimal parameter settings for the intelligent prefetching technique proposed.

4.2.1 Effectiveness of info–stations

In this experiment we ran a number of trips for different info–station configurations with a fixed cache size and prefetch amount. The results are depicted in Figure 7. The figure clearly indicates that the response time is greatly improved with the increasing density of info–stations. When the info–station covered area changes from 0% to 20%, the average response time is improved by more than two fold. If the density increases to approximately 30%, the average response time drops to about 500msec, which is perceivably much more acceptable than the 4.3sec average of no info–stations. Further increase of info–station density continues to improve response time, but at a lot slower rate. This proves that we don't need a large number of info–stations to achieve great benefit to applications. Section 4.2.4 gives a more detailed study of the info–station topology parameters.

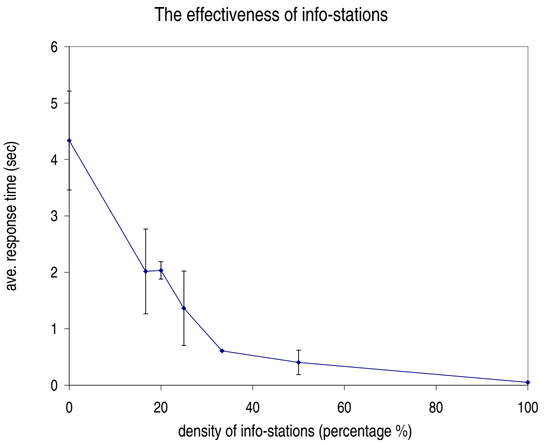


Figure 7: This graph demonstrates response time decreasing significantly with a change from no info–station to a small info–station density. The local driving scenario is used here. The client is in coverage of a info–stations for 2 minute each time. It uses 2MB cache size and prefetches 100KB after each request. See Section 4.2.1 for more details.

4.2.2 Does the cache size matter ?

The cache size is an important design parameter for mobile applications. It is therefore crucial to understand how it influences their performance. We originally expected the cache size to dominate the performance, since intuitively the more data that can be stored, the better the chances a request will hit in the cache.

On the contrary, we find that the cache size does not affect the performance very much. First, with no prefetching involved, Figure 8 shows the average response time with standard deviation under varies of cache sizes. We see no sign of performance increase with larger cache sizes.

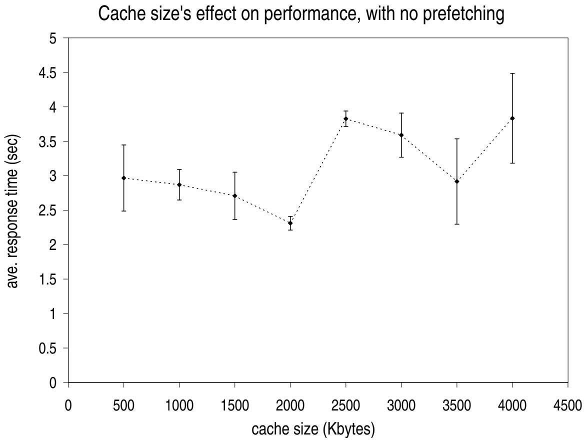


Figure 8: The response time varies indefinitely across cache sizes, with no prefetching. See Section 4.2.2 for more details.

To correlate the effect of prefetching, Figure 9 shows the average response times for different prefetching amounts (100KB, 200KB and 300KB) over a variety of cache sizes. We see also each curve shows greatly varying response times across all cache sizes.

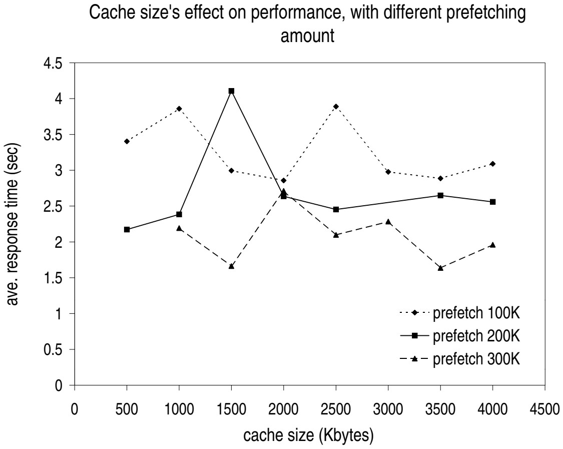


Figure 9: The response time varies indefinitely across cache sizes, with some fixed prefetching amount. Error bars are not plotted here to ease illustration. See Section 4.2.2 for more details.

This in fact follows directly from the user behavior. A user driving about her route exposes very little locality, i.e., the pages touched are unlikely to be touched over and over again. Most of the time a driver will not request the map segments of the places she has already passed, except for the cases when she gets lost and backtracks which occurs only once or twice in every experiment run.

4.2.3 Prefetching strategy

It seems intuitive to assume that the more data that is prefetched, the better performance the application exhibits. However, prefetching large amounts of data is expensive, especially in wireless environments where every bit transferred costs time and money. We therefore seek to understand how different prefetching strategies behave and whether there exists an optimal amount to prefetch given the application and the user behavior.

In these experiments we observe that the average response time decreases with increasing amount of data prefetched, as depicted in Figure 10. This is expected behavior. What is interesting is that after the prefetching amount exceeds 400KB, the response time improvement is not noticeable to the user. This suggests an upper limit to prefetching amount.

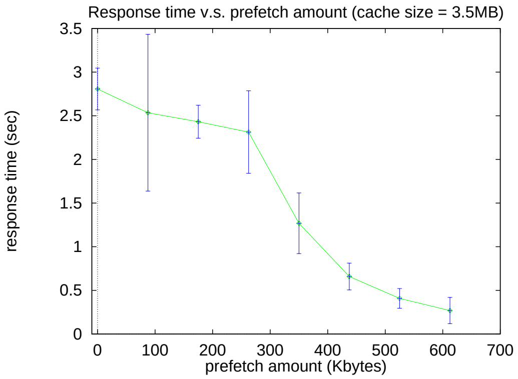


Figure 10: The response time decreases with increasing prefetching amount. A naive prefetching strategy is used. The cache size used is 3.5Mbytes. See Section 4.2.3 for more details.

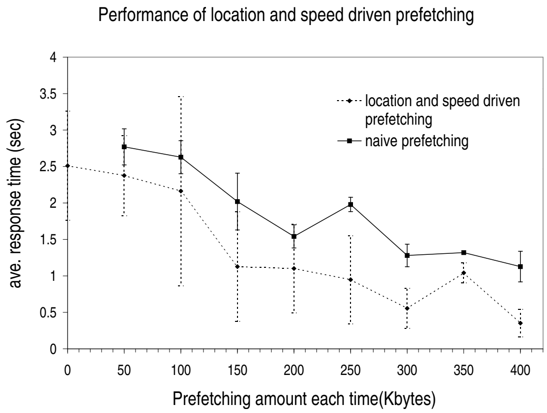


Figure 11: Comparison between the location and speed driven prefetching and a naive prefetching algorithm in terms of response time. Location and speed driven prefetching clearly results in better performance. See Section 4.2.3 for more details.

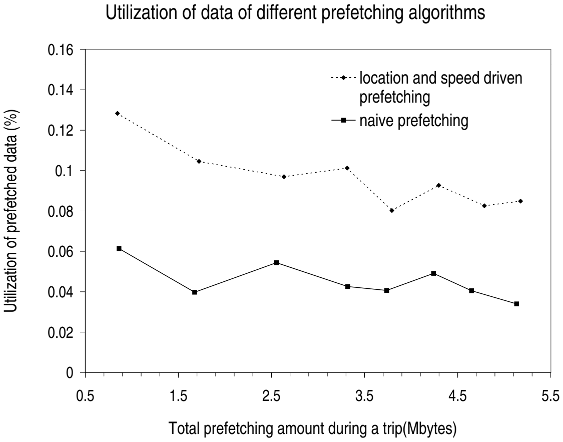


Figure 12: Comparison between the location and speed driven prefetching and a naive prefetching algorithm in terms of data utilization. See Section 4.2.3 for more details.

We now compare the intelligent (location and driving speed dependent) prefetching algorithm with a naive algorithm that always prefetches as much as possible, i.e., the map segment with all its detail levels. Intelligent prefetching performs substantially better than the less adaptive prefetching algorithm. The average response time graph is plotted in Figure 11. The latency improvement is between 16% to 50%, depending on the prefetched amount. The other aspect of performance win, lower cost of our intelligent prefetching, is demonstrated in Figure 12 which plots the utilization of the prefetched data by the application for different prefetching schemes. Although the overall utilization is fairly low, we see that for the same utilization rate, a lot less data have to be prefetched under our algorithm, which leads to lower cost. How to improve the overall data utilization with more knowledge of application will be a challenge for new prefetching algorithms.

4.2.4 Benchmarking the topology of info–stations

The Map–on–the–Move application is used to benchmark the layout of the network of info–stations. We seek to determine the most cost effective way of laying out the info–stations within the overall network to maximize application performance.

As described in the model section (Section 3.3), we use the length of time a mobile client stays in an info–station (at one fixed driving speed) to express `coverage`, i.e., a 2 minute info–station network is a network where a client is covered by each info–stations for 2 minutes when driving at constant speed. `Density` is calculated as the percentage of time spent in info–stations over the entire time in the network.

As shown in Figure 13, in a local driving scenario (with a fixed speed), we can see that the 1 minute info–stations perform significantly better than 2 minute info–stations when distributed with the same density. This means having many info–stations that cover small ranges is a more optimal topology than having few info-stations that cover large ranges. This is only a qualitative analysis because we did not consider signaling delays. But the result shows that we could use many relatively small and inexpensive info–stations to achieve good performance. This corresponds well to the micro-cell argument by [Met95]. If this were constructed, too frequent handoff might yield a performance problem, as in today's micro–cell networks.

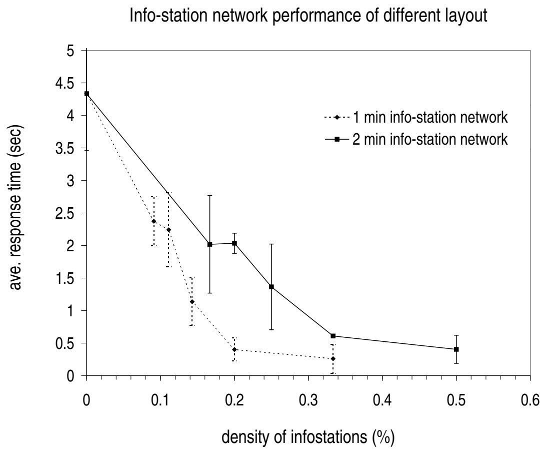


Figure 13: The network topology with info–stations that cover 2 minute range are compared with that of 1 minute. Note we fix user's driving speed in this experiment. With the same density, the 1 min info–stations performance substantially better. See Section 4.2.4 for more details.

These interesting results show us the characteristics of info–stations and our mobile–aware application's interaction with the info–station network. We conclude that info–station networking with application prefetching greatly improved the performance. Our location and speed dependent prefetching algorithm works more efficiently than a normal prefetch-as-much-as-possible algorithm. The benchmark results show small but frequent info–station coverage is more favored.

5 Related Work

The idea to deploy "information islands" along roads to enable wireless communication for driving vehicles has first been studied by Take *et al.* [TMOIT94b, TMOIT94a]. They propose a mobile packet communication system based on chained isolated microradio zones and study the relationship between the time a vehicle spends in a radio zone and the achievable throughput. In their model zones are deployed isolated, whereas we experiment with a system that assumes connectivity everywhere, but non–uniform.

Infostations have first been mentioned by Imielinski [Imi96] in a DataMan project perspective. Frenkeil *et al.*, [FI96], introduced the notion of an *'Infostation'*, as an 'isolated, low cost, low power, short range base station', placed at 'predictable accessible locations', such as toll booths, building entrances, or airport lounges [FI96] [2]. Frenkeil and Imielinski, [FI96], sketches the conceptual design of an *Infostation*, its impact on future PCS, and the potential differences in design of the OSI layers, as required by mobile versus stationary clients. However, no attribute to mobility–awareness within the application design is given.

Network variation awareness has been addressed heavily from the system level. Snoop TCP [BSAK95] and indirect TCP [BB95] propose transport layer protocol improvement for wireless networks. Fox *et al.*, [FGBA96], showed how proxy support can be used in wireless networks. In our work, we demonstrate how to improve performance from the application level. Other approaches, such as the Rover toolkit [JTK96], investigate the possibility of disconnected operation but do not emphasize high mobility as under driving condition. The Odyssey [NSN$^+$97] project shows in general how to provide effective operating system support for concurrent execution of diverse mobile applications over a network with unpredictable bandwidth variation, while our approach focuses on utilizing predictable bandwidth change for application design.

The utility of wireless data access to documents and maps in the field has been discussed by Morey [Mor97] who studies the tradeoff between different delivery and retrieval techniques on a conceptual basis. Discussion of services and market possibilities based on the availability of navigable digital map databases in Europe and the US can be found in [Ess94, Bas96] and [Mor97]. To the best of our knowledge, no other approach has discussed the explicit application of incremental map downloading for mobile clients.

Related work is discussed by Schilit and Theimer [ST94] who address the problem of object location information distribution in local areas, such as buildings and campuses. Their aim, however, is to provide a single "active map service" for the entire environment that keeps track of location and characteristics of objects. We are investigating the problem of geographical map downloading as mobile clients move about the environment, especially through regions with alternating communication characteristics.

Caching and prefetching techniques have been successfully employed to alleviate user perceived latencies for a long time. Prefetching has been widely studied and applied to improve system performance in different areas ranging from software system design to hardware design, e.g., in database systems [Sto81], operating systems [PGS93], in language compilers [Tri79], and microprocessors design [CB92] have prefetching techniques been successfully employed.

Techniques referred to as stashing and hoarding are used in mobile computing environments [KS92] and [TLAC95]. These approaches are very similar to prefetching. Their focus is more on improving availability of the data in the system as opposed to directly influencing performance gain.

The intelligent prefetching algorithm developed in this work is *user–driven* in the sense that hints, provided by the user, are applied to derive prefetching decisions. We are explicitly incorporating route, location, and speed dependent information, gathered from the environment, into the prefetching algorithm. These hints are used to predict the user's future reference needs. This more exact information eases request prediction and consequently improves performance (i.e., decreases user perceived latency).

6 Future Work

Our results are promising and provide a rich foundation for further exploration. We plan to extend this work by additionally incorporating a push–based server model which will allow us to quantitatively study the tradeoff between push and pull–based information retrieval in wireless networks, as has recently been proposed qualitatively by Franklin [Fra97].

Presently we assume that an info–station is primarily deployed to implement the idea of a high bandwidth, low

[2] Note, the release date of this communication is 1996. However, it was not permitted to be released to the general public until 1997 (See attached note to [FI96]).

latency region in the wide–area wireless network. This concept can certainly be extended. Future studies can attribute more functionality to an info–station, e.g., database, separate application caches, user proxies, filtering and data compression.

Furthermore, we think that, due to the user–specific nature inherent in many location dependent applications, an access–history driven prefetching strategy, based on machine learning techniques, could be of great benefit. The application could automatically adapt to specific user access patterns and thus customize itself to changing user requirements.

Future work can also go into an analytical model of the mobile client–server scenario which incorporates the different caching and prefetching strategies and the network layout. The model will help to capture network deployment cost, as well as, the cost incurred by the mobile client. It can function as a tool for the info–station network designer to help find the optimal network layout.

7 Conclusion

To achieve high bandwidth coverage in wide–area wireless networks we have experimented with an alternative network layout based on localized regions of high bandwidth and low latency (info–stations). These regions provide high connectivity to interacting applications for a short period of time. This alternative network layout, and the mobile character of many emerging applications of such networks, demand for a mobility–aware application design. We propose an architecture for implementing a mobility–aware client–server application that takes advantage of the network of info–stations by changing its data–retrieval pattern intelligently according to the distribution of the high bandwidth regions. Our experiments show that smart network layout, combined with mobility–aware application design, can greatly reduce user perceived latency for the class of location dependent applications we have investigated. We show that intelligent prefetching, the technique we developed to improve the mobile-awareness of the application, greatly reduces user perceived latency. Our location and speed driven prefetching algorithm reduces latency by 16% to 50% as compared with a naive *prefetch–as–much–as–possible* algorithm. Furthermore, we found by benchmarking different network topologies, that small but frequent info–station coverage is better suited for the applications studied then other network topologies.

Acknowledgement

We thank Steve McCanne for his advice. We also thank Tom Henderson and Emile Sahouria for their help in extending the network simulator with a client–server model.

This work was sponsored through DARPA DAAB07-95-C-D154 (BARWAN project) and DaimlerBenz Research and Technology North America through the California State MICRO Program. Ye was sponsored by a National Science Foundation fellowship. Jacobsen was supported by the German Research Society, Berlin–Brandenburg DFG grant no. GRK 316.

References

[AFZ96] S. Acharya, M. Franklin, and S. Zdonik. Disseminating updates on broadcast disks. In *22nd International Conference on Very Large Data Bases (VLDB 96)*, Bombay, India, Sep 1996.

[Bad96] B. R. Badrinath. To send or not to send: Implementing deferred transmission in a mobile host. In *DCS'96*, Hong Kong, May 1996.

[Bad97] B. R. Badrinath. Infostations. http://athos.rutgers.edu/˜ badri/dataman/ info.html, 1997.

[Bas96] A. Bastiaansen. The navigatable digital street map is the critical success factor for vehicle navigation and transport information systems in europe. In *IEEE Intelligent Vehicles Symposium*, Sep 1996.

[BB95] A. Bakre and B. R. Badrinath. Indirect tcp for mobile hosts. In *Distributed Computing Systems*, May 1995.

[BGR+98] E. Berruto, M. Gudmundson, R.Menolascino, W. Mohr, et al. Research activities on umts radio interface, network architectures, and planning. *IEEE Communications Magazine,*, 36(2):82–95, Feb 1998.

[BSAK95] H. Balakrishnan, S. Seshan, E. Amir, and R. Katz. Improving tcp/ip performance over wireless networks. In *Mobile Computing and Networking*, Hong Kong, Nov 1995.

[BW96] B. R. Badrinath and G. Welling. Event deliver abstraction for mobile computing. Technical report, Brown University, 1996.

[CB92] T. V. Chen and J. L. Baer. Reducing memory latencies via non–blocking and prefetching caches. In *ASPLOS–V*, Oct 1992.

[CP93] M. Chelouche and A. Plattner. Mobile broadband system (mbs): trends and impact on 60 ghz band mmic development. *Electronics & Communication Engineering Journal*, 5(3):187–97, June 1993.

[Dag88] J. Dagpunar. *Principles of Random Variable Generation.* Oxford Science Publications, 1988.

[Ess94] R. J. Essen. Realization of the european digital road map from concepts to commercialization. In *First World Congress on Applications of Transport Telematics and Intelligent Vehicle-Highway Systems*, 1994.

[FGBA96] A. Fox, S. Gribble, E. Brewer, and E. Amir. Adapting to network and client variability via on–demand dynamic distillation. In *Architectural Support for Programming Languages and Operating Systems*, Cambridge, MA, May 1996.

[FI96] R. H. Frenkeil and T. Imielinski. Infostations: the joy of many-where, many-time communication. Technical Report 119, Winlab, Rutgers University, April 1997 (1996). Winlab Proprietary '*From one year from the date* *of this document (4/1996), distribution limited to Winlab personel; members of Rutgers Administration; and Winlab sponsors, who will distribute internally when appropriate for their need.'*.

[Fra97] M. J. Franklin. Push vs. pull. Talk given in DBLUNCH at UC Berkeley, may 1997.

[FZ96] M. J. Franklin and S. Zdonik. Dissemination-based information systems. *IEEE Data Engineering Bulletin*, 19(3), Sep 1996.

[Hal95] G.A. Halls. Hiperlan-the mbit/s radio lan. In *IEE Colloquium on 'Radio LANs and MANs*, pages 1/1–8, London, UK, April 1995.

[ICG+97] Irvine, J.-P. Couvy, F. Graziosi, J. Laurila, et al. System architecture for the mostrain project (mobile services for high speed trains). In *1997 IEEE 47th Vehicular Technology Conference. Technology in Motion*, volume 3, pages 1917–21, Phoenix, AZ, USA, May 1997.

[Imi96] T. Imielinski. Mobile computing: Dataman project perspective. In J.C. Baltzer, editor, *Mobile Networks and Applications*. AF, Science Publishers, 1996.

[JP94] B. Julich and D. Plassmann. Protocol design and performance analysis of an intermediate-hop radio network architecture for mbs. In *Wireless Networks - Catching the Mobile Future - 5th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, volume 4, pages 1178–82, 1994.

[JTK96] A. D. Joseph, J. A. Tauber, and M. F. Kaashoek. Building reliable mobile-aware applications using the rover toolkit. In *2nd International Conference on Mobile Computing and Networking (MobiCom'96)*, Nov 1996.

[KLO97] A. Koutsaftiki, P. Lane, and J.J. O'Reilly. Proposals on umts system configuration. In *IEEE 6th International Conference on Universal Person Communications Record. Bridging the Way to the 21st Century, ICUPC '97*, volume 1, pages 332–6, 1997.

[Kru93] J. Kruys. Hiperlan, applications and requirements. In *European Optical Communications and Network*, pages 104–8, Geneva, Switzerland, 1993.

[KS92] J. Kistler and M. Satyanarayanan. Disconnected operation in coda file system. *ACM Trans. of Computer Systems*, 10(1):3(25), Feb 1992.

[Map96] Mapquest. *Mapquest*, 1996. http://www.Mapquest.com.

[Mcc] S. Mccanne. Network simulation – ns. http://www-mash.cs.berkeley.edu/ns.

[Met95] Metricom Inc. *Ricochet Network*, 1995. http://www.ricochet.net.

[Mor97] M. E. Morey. Mass storage for delivery of maps and documents to the field. In *IEEE Colloquium on Engineering the Benefits of Geographical Information Systems*, Feb 1997.

[NSN+97] B. D. Noble, J. Satyanarayanan, D. Narayanan, J. Tilton, J. Flinn, and K.R. Walker. Agile application–aware adaptation for mobility. In *1997 16th ACM Symposium on Operating System Principles*, 1997.

[O'M98] D. O'Mahony. Umts: the fusion of fixed and mobile networking. *IEEE Internet Computing*, 2(1):49–56, 1998.

[PGS93] H. Patterson, G. Gibson, and M. Satyanarayanan. A status report on research in transparent informed prefetching. *SIGOPS, Operating Systems Review*, 27(2):21–34, April 1993.

[ST94] B. N. Schilit and M. N. Theimer. Disseminating active map information to mobile hosts. *IEEE Network*, pages 22–32, Sept Oct 1994.

[Sto81] M. Stonebraker. Operating system support for database management. In *Communication of the ACM*, volume 24, July 81.

[TLAC95] C. Tait, H. Lei, S. Acharya, and H. Chang. Intelligent file hoarding for mobile computers. In *ACM Conf. on Mobile Computing and Networking*, Berkeley, CA, Nov 1995.

[TMOIT94a] K. Take, Y. Mita, T. Oh-Ishi, and H. Tominaga. A mobile packet communication with a chained isolated radio zone network system. *Transactions of the Institute of Electronics, Information and Communication Engineers B-1*, 1(6):405–13, June 1994. (in Japanese).

[TMOIT94b] K. Take, Y. Mita, T. Oh-Ishi, and H. Tominaga. A placing method of micro radio zones in a mobile packet communication with a chained isolated radio zone network system. *Transactions of the Institute of Electronics, Information and Communication Engineers B-1*, 1(6):414–23, June 1994. (in Japanese).

[Tri79] K. Trivedi. An analysis of prepaging. *Computing*, 3(22), 1979.

[Wat94] T. Watson. Application design for wireless computing. In *IEEE Workshop on Mobile computing*, Dec 1994.

[XII96] European Comission DG XIII/B. Umts task force report. Technical Report 1, European Comission, Brussels, March 1996.

[YJ97] T. Ye and H.-A. Jacobsen. Mobile awareness in a wide area wireless network of info–stations. Computer Networks graduate class project report Computer Science Division, UC Berkeley, May Spring 1997.

[Zub94] J.T. Zubrzycki. Mbs-a wireless network for digital video. In *International Broadcasting Convention (Conf. Publ. No.397)*, pages 266–71, Amsterdam, Netherlands, Sept. 1994.