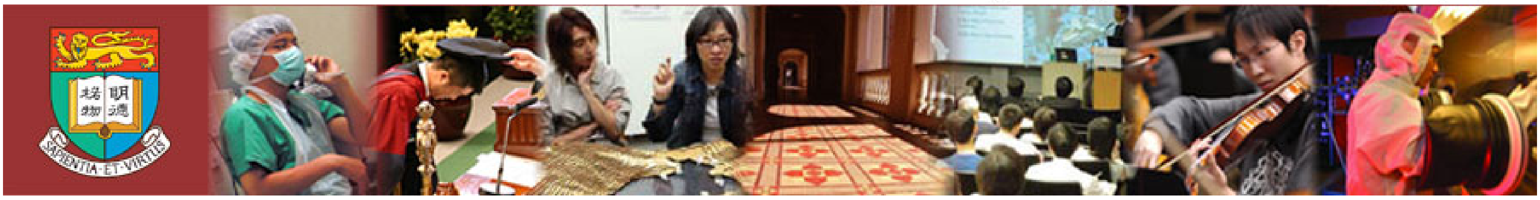


Design and Analysis of Channel Adaptive Wireless Cache Invalidation Strategies with Downlink Traffic

Mark Kai Ho Yeung and Yu-Kwong Kwok
Department of Electrical and Electronic Engineering
The University of Hong Kong, Pokfulam Road, Hong Kong
Corresponding Author: Yu-Kwong Kwok (email: *ykwok@hku.hk*)

Abstract— **In this paper, we study the performance of the IR+UIR wireless data cache invalidation approach under a realistic system model: the quality of the wireless channel is time-varying; and there are other downlink traffics in the system. Our analysis and simulation results show that query delay significantly increases as a result of broadcast error and the additional downlink traffics experience longer delay due to extended broadcast period. Exploiting link adaptation (i.e., transmission rate is adjusted dynamically according to channel quality), we then propose three schemes to tackle these two problems. Our results indicate that the proposed schemes outperform IR+UIR under a wide range of system parameters.**

Keywords: cache invalidation, invalidation report (IR), link adaptation, channel adaptive protocols, wireless networks, client-server computing, system design, simulations.

I. Introduction

The basic wireless cache invalidation strategy is the use of *invalidation reports* (IRs) [1]. Server periodically broadcasts IRs, each of which indicates those data items that are recently updated at the source. Clients use IRs to keep their caches consistent by discarding any obsolete data. If a query cannot be served locally, i.e., a cache miss, the client issues an uplink query request for the data item. The server aggregates query requests from all its clients and broadcasts query replies only once every IR broadcast period. In this manner, a query reply can be shared by more than one client via the broadcast channel. The major advantages of this cache invalidation strategy are high scalability and energy efficiency [4]. First of all, the size of IR is independent of the number of clients. Secondly, clients can exploit the periodicity of server broadcast to save power in that mobile devices can operate in doze mode most of the time and only become active during server broadcast.

However, if the disconnection time of a client is longer than a fixed period of time, the client should discard its entire cache even if some of the cached data may still be valid. This issue is addressed in [1], [7], [8], [16]. Another drawback common to IR-based approaches is the large query delay involved since clients require an IR to ensure cache consistency. The addition of updated invalidation reports (UIRs) to the original IR-based approach (IR+UIR) [2], [3], [4] aims to tackle this problem. In IR+UIR, server broadcasts a number of UIRs between successive IRs. Specifically, each UIR only contains information about most recently updated data since the last IR. As a result, the use of UIRs requires the cache is consistent up to the last IR. Clients use UIRs to keep their cache consistent to the source. Thus, for cache hits, there is no need to wait for the next IR and query delay is further reduced.

Unfortunately, previous research results on wireless cache invalidation are based on two simplifying assumptions: (i) the broadcast channel is error-free; (ii) there is no other downlink traffic in the system. As such, the applicability of previous approaches under a realistic environment, where the above assumptions are usually invalid [11], is largely unknown. Indeed, the first assumption is clearly unrealistic as signal propagation impairments [12] (e.g., multipath fading due to user mobility, long term shadowing due to terrain configurations, interferences due to other wireless devices, etc.), and hence, packet reception failures, are inevitable in a practical situation. The second assmuption is also inapplicable in real life because mobile devices are usually used for multiple purposes at the same time (e.g., a mobile phone equipped with a browser may be used for Web-surfing while having a phone conversation). In this paper, based on mathematical analysis (see Section III) and extensive simulations (see Section IV), we present our study on how the IR+UIR approach performs under such a realistic environment, in which other downlink traffic exists and a *channel adaptive* physical layer is incorporated (a commonly used facility in many practical wireless systems, such as UMTS, IEEE 802.11x WLAN, etc. [5]). Link adaptation is an effective strategy [6], [10] which works by dynamically adjusting the transmission rate through varying the amount of forward error correction (FEC) bits included in a packet, according to the channel quality.

First, we find that the performance of the IR+UIR algorithm critically depends on the integrity of IRs. Second, fixed-rate broadcast (i.e., the channel adaptation facility is not used) is clearly undesirable in that it leads to either (i) significant under-utilization of the wireless channel; or (ii) causing severe transmission errors. Furthermore, despite that IR+UIR improves query delay by minimizing the delay in cache hits, the penalty due to cache miss is still large. In view of these problems, we propose three different schemes (see Section II) for enhancing the cache invalidation strategies. Our simulation results indicate that our proposed schemes can reduce: (i) the effect of broadcast error to IR+UIR; and (ii) the impact of broadcast overhead on other downlink traffic.

This research was supported by a grant from the Hong Kong Research Grants Council under project number HKU 7162/03E.

II. Proposed Cache Invalidation Strategies

In this section, we describe our three proposed cache invalidation strategies targeted at: (i) reducing the probability of corruption in IRs; (ii) improving the broadcast channel utilization; (iii) reducing the average delay in other downlink traffic, respectively.

A. System Architecture

In our system model, the base-station is connected to a database via a wired network. Each base-station (server) serves the users within its covered area (cell). There are two types of users in the system, namely: voice and data. A voice session requires r_{speech} Kbps. Each data user (client) is making a file transfer session at r_{file} Kbps and also generating a stream of exponentially distributed read-only requests with mean arrival time T_q. The database is divided into hot and cold sets. Updates to each data item follow an exponential distribution with mean arrival time T_u, from which hot items have an update probability of p_u. We assume that each server has a consistent view on the common database and broadcasts the same set of IRs and UIRs. Thus, it suffices to maintain cache consistency among all clients within one cell. As in most previous works [4], [7], [13], [15], we adopt the latest value consistency model, i.e., the most recent value of a data item is used to serve a query. Apart from the server and clients, in our approach another entity is introduced: the broadcast scheduler, which determines, from the set of available transmission modes with different levels of error protection (see [9], [14]), the transmission rate for the broadcast traffic (see Section II-C).

B. Proposed Scheme 1: Reducing the Probability of Corruption in IR

Server broadcasts invalidation reports (IRs) every L seconds. Each IR contains the current timestamp (T_i) and a list of (d_x, t_x) pairs, where d_x is the id of a data item and t_x is the timestamp at which the data item is last updated. To support long disconnection period, data items that are updated in the past ωL seconds are included in each IR, i.e., $t_x > (T_i - \omega L)$, where ω is the broadcast window size. The server also broadcasts $(m-1)$ updated invalidation reports (UIRs) between successive IRs, where $(m-1)$ is the UIR replicate times. Each UIR indicates those data items that are updated since the last IR. Clients make use of IRs and UIRs to keep their caches consistent by discarding obsolete data such that some queries can be served locally. However, the use of UIRs requires the cache is consistent up to the last IR. If a client misses the last IR, subsequent UIRs before the next IR cannot be used. Thus, the effectiveness of UIRs depends on the integrity of each IR.

Depending on the update arrival rate to the database, an IR may include a large number of id-timestamp pairs. It is crucial to ensure that clients can successfully receive an IR. However, broadcasting the entire IR at once over the error-prone wireless channel is inefficient due to the fact that such a long transmission is highly vulnerable to channel errors, especially for those clients with particularly unfavorable channel conditions (e.g., with a high mobility or situates in a poor terrain). Since there may be a large number of clients listening to the same IR in the broadcast channel, it is impractical for the server to retransmit an IR to completely avoid corruption. Instead, to reduce the chance of corruption, an original IR is divided into ω segments, IR$_j$ for $j = 1$ to ω. Each IR segment then contains only id-timestamp pairs in a represented broadcast period, i.e., (d_x,t_x) is included in IR$_j$ if and only if $(T_i - jL) \leq t_x < (T_i - (j-1)L)$, for $j = 1$ to ω. The packet overhead increases from one to ω. However, if an id-timestamp pair is included in the original IR, it appears in one and only one IR segment. The total number of id-timestamp pairs in the ω segments remains the same as that in the original IR.

Algorithm 1 IR Partitioning

```
INPUT: $D$ (the set of data items); $d_x$ (id of a data item); $t_x$ (last update timestamp); $L$ (broadcast interval); $\omega$ (broadcast window); $T_i$ (the current broadcast time);
OUTPUT: IR$_j$ (the $j$th IR segment);

for j = 1 to $\omega$ do
    IR$_j$ = $\{\langle d_x, t_x \rangle | (d_x \in D) \wedge (T_i - jL < t_x \leq T_i - (j-1)L)\}$;
    Broadcast IR$_j$;
end for
```

Algorithm 2 IR Reconstruction

```
INPUT: $L$, $\omega$, $T_i$; $T_{\text{last}}$ (timestamp of the last correct IR); $IR_{\text{last}}$ (last correct IR);
OUTPUT: $IR'$ (IR at $T_i$);

$j = \mathbf{min}\,\{\,\omega, \frac{T_i - T_{\text{last}}}{L}\}$ (since there are only $\omega$ segments)
if all the latest $j$ segments at $T_i$ are uncorrupted then
    $IR'$ = Reconstruct(IR$_{\text{last}}$,all the latest $j$ segments);
    $T_{\text{last}} = T_i$;
    Normal operation using $IR'$;
else
    Wait for the next IR at $T_i + L$;
end if

Function Reconstruct(IR$_{\text{last}}$, all the latest $j$ segments)
$IR' = \emptyset$;
for each $(d_x, t_x)$ in the latest $j$ segments do
    $IR' = IR' \cup (d_x, t_x)$;
end for
for each $(d_x, t_x)$ in $IR_{\text{last}}$ do
    if $(d_x, t_x) \notin IR'$ then
        $IR' = IR' \cup (d_x, t_x)$;
    end if
end for;
Return $IR'$;
```

Algorithm 1 formalizes how the server constructs and broadcasts the IR segments. If a client's cache is consistent up to T_{i-1} (using IR at T_{i-1}), the client is required to retrieve only the latest segment at T_i, i.e., IR_1. The complete IR at T_i can be reconstructed locally using IR_1 at T_i and IR at T_{i-1}. Similarly, if a client misses both the IRs at T_{i-2} and T_{i-1}, then IR at T_{i-3} and the latest two IR segments at T_i are required for local reconstruction. Algorithm 2 formalizes how the local reconstruction process is performed at the client.

C. *Proposed Scheme 2: Improving Channel Utilization*

To improve the success probability of broadcast traffic, the optimal transmission rate should depend on (i) the current channel status of all clients, and (ii) the importance of the information being delivered. This idea is illustrated as follows: deliver more important information using a low-rate (i.e., higher level of error protection), to improve the success probability; deliver less important information using a high-rate (i.e., lower level of error protection), to better utilize the wireless channel.

For active clients, the latest IR segments (e.g., IR_1, IR_2, IR_3), UIRs and query replies are necessary for correct operations: maintain cache consistency and serve queries. Thus, they are more appropriate for low-rate broadcast to provide better protection against transmission errors. The other IR segments, which contain old cache invalidation information, are only necessary for some clients that have been disconnected for some period of time ($< \omega L$). As such, these old segments are more suitable for high-rate broadcast to better utilize the channel. However, low-rate and high-rate broadcasts are not fixed. What transmission rates are suitable for low-rate and high-rate broadcast is a function of the channel conditions of all clients. A broadcast scheduler is introduced at the server side to determine the optimal transmission rates for different broadcast traffic. The broadcast scheduler collects the channel status information (CSI) from all clients with the aid of the polling subframe in the downlink and pilot subframe in the uplink. As discussed above, it is not good to use either an aggressive (i.e., high rate) or a conservative (i.e., low rate) broadcast strategy. Thus, the average data rate, computed based on all CSIs, is used to indicate the aggregate channel conditions of all clients. The average rate is used to determine the transmission mode in broadcasting the more important information. From simulation results, it is observed that the latest three IR segments are more important to mitigate the effect of transmission errors. The less important one is then transmitted with one higher mode (see [9], [14]). The algorithm at the broadcast scheduler is outlined in Algorithm 3.

Algorithm 3 Broadcast Scheduling

```
INPUT: $CSI_i$ (the instantaneous channel state information of client $i$); $CSI_{\mathrm{mean}}$ (the nominal CSI based on average rate);
OUTPUT: $M_{\mathrm{basic}}$ (basic broadcast mode for better protection); $M_{\mathrm{enhanced}}$ (enhanced broadcast mode for higher transmission rate);

for each downlink frame do
    Calculate $CSI_{\mathrm{mean}}$ from all $CSI_i$;
    $M_{\mathrm{basic}}$ = using $CSI_{\mathrm{mean}}$, lookup the corresponding transmission mode from the set of available adaptive error protection levels.
    $M_{\mathrm{enhanced}} = \mathbf{min}\ \{5, M_{\mathrm{basic}} + 1\}$; (since the highest mode is 5)
    if the current information slot carries data for (i) the latest 3 IR fragments ($IR_1$, $IR_2$ and $IR_3$) or (ii) UIRs or (iii) query replies then
        Use $M_{\mathrm{basic}}$ to broadcast;
    end if
    if the current timeslot carries data for the remaining old IR fragments ($IR_4$ to $IR_\omega$) then
        Use $M_{\mathrm{enhanced}}$ to broadcast;
    end if
end for
```

D. *Proposed Scheme 3: Reducing the Average Delay in Other Downlink Traffic*

Depending on the client size and the query generation rate of each client, there may be a long list of query replies following each IR. As discussed in Section II-B, the size of each IR can be very large in order to support long disconnection period. These two factors cause the broadcast traffic to occupy the channel for a long time. As a result, other downlink traffic would experience a large delay. The severity of the large delay depends on the nature of other downlink traffic. For elastic traffic, there is little harm suffering from slightly larger delay. However, large delay is unacceptable for delay sensitive traffic such as real-time voice or video. To reduce the adverse effect of broadcast on other downlink traffic and the query delay experienced by all clients, the server is to broadcast query replies after both IRs and UIRs instead of just IRs. This would shorten the longest broadcast time, which reduces the average waiting time of other downlink traffic spent at the server. The proposed scheme can also improve the query delay by reducing the penalty due to cache miss.

After receiving an UIR and invalidating any obsolete data items, the client notices that a query cannot be served locally, i.e., cache miss. The client then issues an uplink query request to the server at T_q. In the IR+UIR scheme, the query reply is scheduled to be broadcast after the next IR at T_i. Equivalently, the expected query delay due to cache miss is half the IR broadcast period, i.e., $\frac{L}{2}$ seconds. In our proposed scheme, however, the expected delay for a cache miss is the same as that for a cache hit, i.e., $\frac{L}{2m}$ seconds. The tradeoff is the decrease in aggregate effect due to the shorter broadcast cycle. A broadcast data item is cached by all clients. When a query for the data item arrives at a later time, depending on the status of the cache, clients will have different actions. This leads to two conflicting effects: If the cached copy is still valid, the query can be served locally and the client can conserve battery power in uplink transmission; otherwise, an uplink request will be issued and the resources expended in previous broadcast of the data item are wasted. Which effect outweighs the other depends on the update arrival rate, query distribution in the database, and also the cache size of each client. Our simulation results in Section IV suggest that our proposed scheme achieves an overall performance improvement in different settings.

III. Performance Analysis

In this section, we present a mathematical analysis of the efficacy of the three proposed schemes. For the complete analysis, please refer to the detailed report [14]. Table I lists the notation used throughout the analysis.

TABLE I
Definitions of notation.

Notation	Definition
D_h	number of data items in the hot data set
D_c	number of data items in the cold data set
r_h	uplink request arrival rate of a hot data item
r_c	uplink request arrival rate of a cold data item
μ_h	update arrival rate of a hot data item
μ_c	update arrival rate of a cold data item
N	total number of clients
L	broadcast interval
ω	broadcast window
$m-1$	UIR replicate times
$S_{\rm data}$	data item size
x	packet overhead
$U_{\rm id}$	unique number size
CSI_i	current channel state information of client i
q	transmission mode ($=0,1,\ldots,5$)
$b(q)$	number of bits in an information slot
P_e	bit error rate (a function of CSI and q)

Proposed Scheme 1: Reducing the Probability of Corruption in IR

Dividing IR into segments increases the broadcast overhead. However, this can reduce the effect of transmission error. It can be shown that the expected size of different cache invalidation components is given as follows:

$$\begin{aligned} S_{\rm IR} &= \text{Expected size of an IR} \\ &= 2U_{\rm id}\left\{D_h[1-e^{-\mu_h\omega L}] + D_c[1-e^{-\mu_c\omega L}]\right\} \\ &\quad +x \end{aligned} \tag{1}$$

$$\begin{aligned} S_{\rm UIR} &= \text{Expected size of all } (m-1) \text{ UIRs} \\ &= 2U_{\rm id}\sum_{i=1}^{m-1}\left\{D_h[1-e^{-\mu_h\frac{iL}{m}}] + D_c[1-e^{-\mu_c\frac{iL}{m}}]\right\} \\ &\quad +(m-1)x \end{aligned} \tag{2}$$

$$\begin{aligned} S_{\rm IR\_Seg} &= \text{Expected size of all } \omega \text{ IR segments} \\ &= 2U_{\rm id}\left\{D_h[1-e^{-\mu_h\omega L}] + D_c[1-e^{-\mu_c\omega L}]\right\} \\ &\quad +\omega x \end{aligned} \tag{3}$$

$$\begin{aligned} S_{\rm Last\_Seg} &= \text{Expected size of the last IR segment} \\ &= 2U_{\rm id}\left\{D_h[1-e^{-\mu_h L}] + D_c[1-e^{-\mu_c L}]\right\} \\ &\quad +x \end{aligned} \tag{4}$$

Hence, the broadcast overheads are $(S_{\rm IR}+S_{\rm UIR})$ and $(S_{\rm IR\_Seg}+S_{\rm UIR})$ for IR+UIR and the proposed scheme, respectively. We use *increase in broadcast overhead* to quantify the cost of the proposed scheme:

$$\text{Increase in broadcast overhead} = \frac{S_{\rm IR\_Seg}+S_{\rm UIR}}{S_{\rm IR}+S_{\rm UIR}} \tag{5}$$

For the effect of transmission error, we assume that there is no forward error correction (FEC) technique employed, which represents the worst case scenario. In IR+UIR, each client requires to receive a complete IR. In the proposed approach, if a client correctly receives an IR, only the last IR segment in the next broadcast period is needed. We define the increase in number of clients that can use subsequent UIRs as the *performance improvement factor*:

$$\begin{aligned} &\text{Performance improvement factor} \\ &= \sum_{i=1}^{N}\left\{[1-P_e(\mathrm{CSI}_i,q)]^{S_{\rm Last\_Seg}}\right. \\ &\quad \left.-[1-P_e(\mathrm{CSI}_i,q)]^{S_{\rm IR}}\right\} \end{aligned} \tag{6}$$

The performance data for proposed scheme 1 is shown in Figure 1(a). There are 50 clients in the system and the server is using transmission mode 0. It is observed that with moderate update arrival time (50–500s), the scheme shows the greatest improvement with only slightly increase in broadcast overhead.

Proposed Scheme 2: Improving Channel Utilization

Consider an information slot of size b bits, where b is a function of the transmission mode, q. The average number of bits received by each client is used as the performance metric:

$$\begin{aligned} &\text{Average number of bits received by each client} \\ &= \frac{b(q)}{N}\sum_{i=1}^{N}[1-P_e(\mathrm{CSI}_i,q)]^{b(q)} \end{aligned} \tag{7}$$

The question is to determine q such that Equation (7) is maximized. From the above expression, it is observed that the optimal value of q should be dependent of the CSI values of different clients. To reduce the complexity at the server, we adaptively choose q in different channel conditions such that:

$$q_{\text{adapt}} = \frac{1}{N}\sum_{i=1}^{N}\text{CSI}_i \qquad (8)$$

Figure 1(b) shows the performance curves for the different fixed rate broadcast schemes and the proposed adaptive scheme. It is observed that the more conservative schemes (mode 0 and mode 1) are more preferable in a poor channel. On the other hand, the more aggressive ones offer higher throughputs when the channel is good. However, there is no fixed rate broadcast scheme that performs well across different settings. The proposed approach takes advantage of the good channel quality by increasing the broadcast rate but provides better protection to transmission errors when the channel deteriorates. Thus, it is more suitable in a time-varying channel than any of the fixed rate broadcast schemes, as illustrated in Figure 1(b).

Proposed Scheme 3: Reducing the Average Delay in Other Downlink Traffic

Query replies broadcast by the server increase the queueing delay in other downlink traffic. For IR+UIR, query replies are broadcast every IR broadcast interval, L; while, for UIR-Reply, server broadcasts query replies every UIR interval, i.e., $\frac{L}{m}$ seconds. Thus, we have,

$$\begin{aligned} H_1 &= \text{Expected number of IR+UIR query replies} \\ &= (S_{Data} + x) \times \{D_h[1 - e^{-r_h NL}] \\ &\quad + D_c[1 - e^{-r_c NL}]\} \qquad (9) \end{aligned}$$

$$\begin{aligned} H_2 &= \text{Expected number of UIR-Reply query replies} \\ &= (S_{Data} + x) \times \{D_h[1 - e^{-\frac{r_h NL}{m}}] \\ &\quad + D_c[1 - e^{-\frac{r_c NL}{m}}]\} \qquad (10) \end{aligned}$$

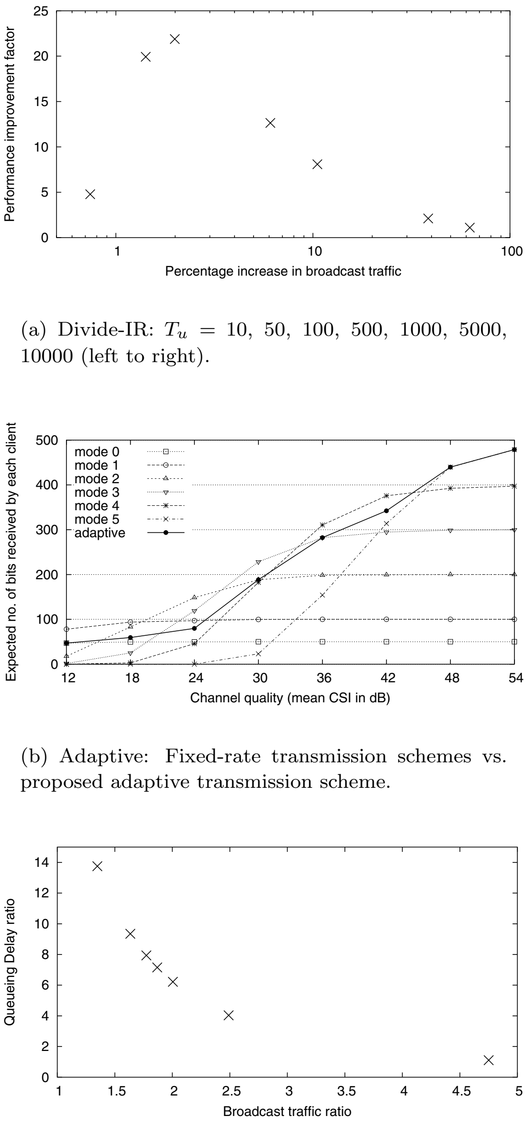
To have a fair comparison, we compare the expected size of query replies over a broadcast interval, i.e., L seconds. In IR+UIR, the expected size is H_1 while, in UIR-Reply, it is $m \times H_2$. From Equations (9) and (10), the ratio of the broadcast traffic in the proposed scheme to IR+UIR is used to quantify the cost of the proposed scheme:

$$\text{Broadcast traffic ratio} = m \times \frac{H_2}{H_1} \qquad (11)$$

When the server is performing broadcast, packets from other downlink traffic are queued for later transmission. The performance gain by the proposed scheme is defined as the *queueing delay ratio*, which is shown to be:

$$\text{Queueing delay ratio} \approx \left(\frac{H_1}{H_2}\right)^2 \qquad (12)$$

The performance data for proposed scheme 3 is shown in Figure 1(c). There are 50 clients in the system and the server is using transmission mode 0. The cache hit ratio is assumed to be 0.8. When queries to the database are very rare ($T_q = 10000$s), the scheme incurs a significant increase in broadcast traffic. Except to this extreme case, the improvement in queueing delay seems justify the increased broadcast traffic.



(a) Divide-IR: T_u = 10, 50, 100, 500, 1000, 5000, 10000 (left to right).

(b) Adaptive: Fixed-rate transmission schemes vs. proposed adaptive transmission scheme.

(c) UIR-Reply: T_q = 10, 50, 100, 500, 1000, 5000, 10000 (left to right).

Fig. 1. Performance data for the three proposed cache invalidation schemes.

IV. Simulation Results

We have also done extensive simulations to compare the performance of various approaches under a wide range of parameters. However, due to space limiations, we can only present a small set of results here. For the complete set of results, the reader is referred to [14]. The effect of mean query generation time is shown in Figure 2. As can be seen, the channel quality, as well as the broadcast scheme, significantly affects the query delay. Furthermore, the improvement by UIR-Reply is more significant in conservative broadcast. This confirms that delay in other downlink traffic is mainly due to the long broadcast time of IRs and data replies. In summary, our proposed schemes perform better than the original IR+UIR by alleviating the effect of transmission errors to the broadcast traffic and the impact of broadcast traffic on other downlink traffic in the system.

References

[1] D. Barbara and T. Imielinski, “Sleepers and Workaholics: Caching Strategies in Mobile Environments,” *Proc. ACM SIGMOD Int’l Conf. Management of Data*, vol. 23, no. 2, pp. 1–12, May 1994

[2] G. Cao, “On Improving the Performance of Cache Invalidation in Mobile Environment,” *ACM Mobile Networks and Applications*, vol. 7, no. 4, pp. 291–303, Aug. 2002.

[3] G. Cao, “Proactive Power-Aware Cache Management for Mobile Computing Systems,” *IEEE Trans. Computers*, vol. 51, no. 6, pp. 608–621, June 2002.

[4] G. Cao, “A Scalable Low-Latency Cache Invalidation Strategy for Mobile Environments,” *IEEE Trans. Knowledge and Data Engineering*, vol. 15, no. 5, pp. 1–15, May 2003.

[5] V. K. Garg, *Wireless Network Evolution: 2G to 3G*, Prentice-Hall, 2002.

[6] A. J. Goldsmith and S.-G. Chua, “Variable-Rate Variable-Power MQAM for Fading Channels,” *IEEE Trans. Communications*, vol. 45, no. 10, pp. 1218–1230, Oct. 1997.

[7] Q. Hu and D. Lee, “Adaptive Cache Invalidation Methods in Mobile Environments,” *Proc. High Performance Distributed Computing*, pp. 264–273, Aug. 1997.

[8] J. Jing, A. Elmagarmid, A. S. Helal, and R. Alonso, “Bit-sequences: An Adaptive Cache Invalidation Method in Mobile Client/Server Environments,” *Mobile Networks and Applications*, vol. 2, no. 2, pp. 115–127, Oct. 1997.

[9] Y.-K. Kwok and V. K. N. Lau, “A Novel Channel-Adaptive Uplink Access Control Protocol for Nomadic Computing,” *IEEE Trans. Parallel and Distributed Systems*, vol. 13, no. 11, pp. 1150–1165, Nov. 2002.

[10] V. K. N. Lau, “Performance Analysis of Variable Rate: Symbol-By-Symbol Adaptive Bit Interleaved Coded Modulation for Rayleigh Fading Channels,” *IEEE Trans. Vehicular Technology*, vol. 51, no. 3, pp. 537–550, May 2002.

[11] D. L. Lee, W.-C. Lee, J. Xu, and B. Zheng, “Data Management in Location-Dependent Information Services,” *IEEE Pervasive Computing*, vol. 1, no. 3, pp. 65–72, July-Sept. 2002.

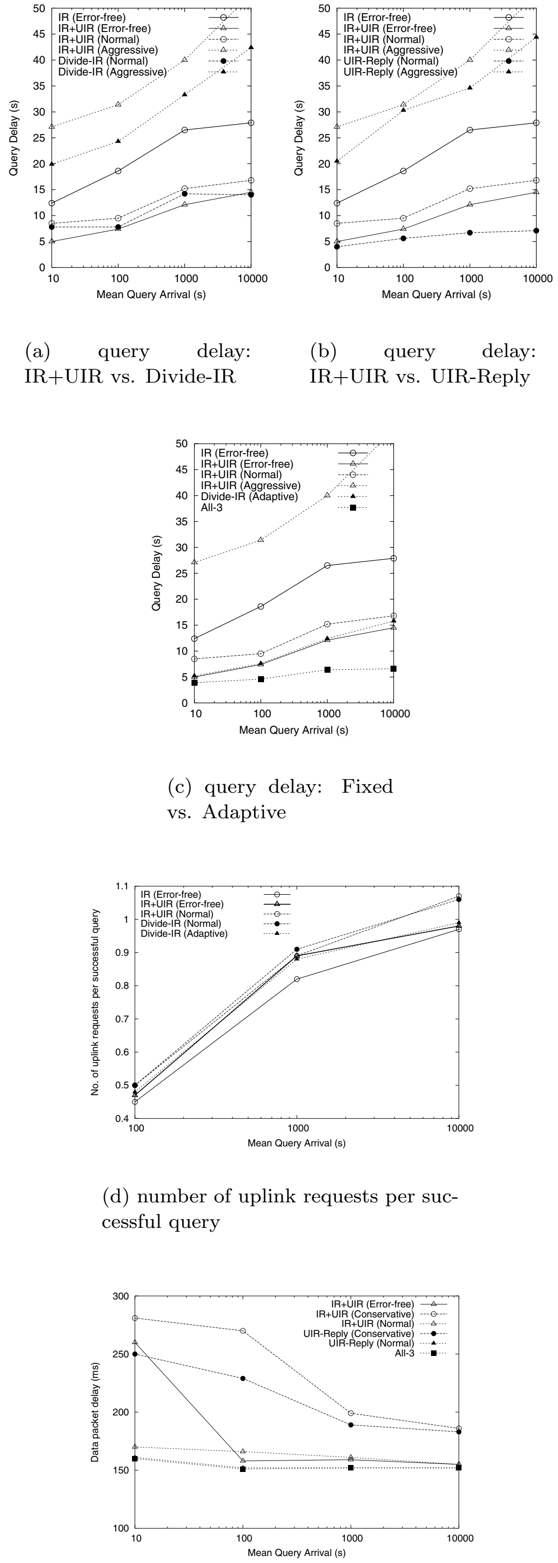
[12] J. D. Parsons, *The Mobile Radio Propagation Channel*, Second Edition, John Wiley & Sons, 2000.

[13] K. Tan, J. Cai, and B. C. Ooi, “An Evaluation of Cache Invalidation Strategies in Wireless Environments,” *IEEE Trans. Parallel and Distributed Systems*, vol. 12, no. 8, pp. 789–807, Aug. 2001.

[14] M. K.-H. Yeung, *On Channel Adaptive Cache Invalidation Schemes in Wireless Client-Server Systems with Heterogeneous Downlink Traffic and Game Theoretic Considerations*, M.Phil. thesis, The University of Hong Kong, Aug. 2004.

[15] L. Yin and G. Cao, “Adaptive Power-Aware Prefetch in Wireless Networks,” *IEEE Trans. Wireless Communications*, accepted for publication and to appear.

[16] B. Zheng, J. Xu, and D. L. Lee, “Cache Invalidation and Replacement Policies for Location-Dependent Data in Mobile Environments,” *IEEE Trans. Computers*, vol. 51, no. 10, pp. 1141–1153, Oct. 2002.



(a) query delay: IR+UIR vs. Divide-IR

(b) query delay: IR+UIR vs. UIR-Reply

(c) query delay: Fixed vs. Adaptive

(d) number of uplink requests per successful query

(e) delay in other downlink traffic

Fig. 2. Effect of mean query generation time (number of clients = 50; T_u = 100s).