A Proxy-Based Integrated Cache Consistency and Mobility Management Scheme for Mobile IP Systems

Weiping He, Ing-Ray Chen, and Baoshan Gu

Department of Computer Science
Virginia Tech
{weiping, irchen, bgu}@vt.edu

Abstract

In this paper, we investigate a proxy-based integrated cache consistency and mobility management scheme in Mobile IP systems. Our scheme is based on a stateful strategy by which cache invalidation messages are asynchronously sent by the server to a mobile host (MH) whenever data objects cached at the MH have been updated. We use a per-user proxy to buffer invalidation messages to allow the MH to disconnect arbitrarily and to reduce the number of uplink requests when the MH is reconnected. Moreover, the MH's proxy serves as a gateway foreign agent (GFA) as in the MIP Regional Registration protocol to keep track of the address of the MH in a region. The proxy migrates with the MH when the MH crosses a regional area. We identify the optimal *regional area size under which the overall network traffic cost, due to cache consistency management, mobility management, and query requests/replies, is minimized. We demonstrate that the integrated cache consistency and mobility management scheme outperforms both no-proxy and/or no-cache schemes in Mobile IPv6 environments.*

1 Introduction

A major challenge for Mobile IP systems is to maintain service continuity even if a mobile host (MH) disconnects and then reconnects at will. A MH may disconnect voluntarily simply to reduce connection cost and/or to avoid power consumption. A MH may disconnect involuntarily due to handoff or wireless link problems [6].

In this paper we consider a Proxy-based Integrated Cache and Mobility Management (PICMM) scheme in Mobile IPv6 networks to support mobile client-server database applications. A MH may query dynamic data such as stock prices, weather reports, or traffic information. To avoid sending a query to the server and receiving a reply through the expensive and often unreliable wireless network, a MH can cache data objects on the local storage and then answer queries for data that are up-to-date. Caching reduces the server access cost and improves user-perceived response time [4].

The scheme proposed in this paper is based on the stateful strategy by which a cache invalidation message is asynchronously sent by the server to the MH, whenever there is an update to a data object cached at the MH. The MH uses invalidation reports received to determine the validity of its cache content before answering a query. If a query asks for a data object that has been invalidated, then a request is sent uplink to the sever to ask for a fresh copy of the data object accessed by the query before the query can be answered. Moreover, to support service continuity in cases the MH is disconnected and then reconnected again, we use a per-user proxy to buffer invalidation messages to allow the MH to disconnect arbitrarily and to reduce the number of uplink requests to check cache status when the MH is reconnected.

The generated network traffic cost associated with cache consistency management thus includes the cost of receiving and buffering invalidation messages at the proxy and the cost of forwarding them from the proxy to the MH, as well as the cost of sending requests to the server and receiving responses in case cached data objects are not up-to-date to answer queries. This cost is considered as part of the "service" management cost which we like to minimize. Another major network traffic cost in MIPv6 systems is due to mobility management to maintain the location of the MH.

To minimize both the "service" and "mobility" network traffic costs, we let the user proxy mentioned above also take the responsibility of mobility management to further reduce the network traffic. We investigate a design by which the MH's proxy will serve as the

MH's gateway foreign agent (GFA) as in the MIP Regional Registration protocol [3, 5] to keep track of the address of the MH in a region. The proxy migrates with the MH when the MH crosses a regional area. This paper aims to identify the *optimal* regional area size under which the overall network traffic generated due to cache consistency management, mobility management, and query requests/replies, is minimized. The idea can be extended to other mobility management protocols such as HMIPv6 [7], IDMP , and HAWAII although in this paper we only consider the Regional Registration protocol. Mtika et al. [5] have analyzed the optimal number of access routers (ARs) in a region in MIPv6 environments. However, they considered the use of regional registers for all MNs and adopted a uniform flow model assuming that all MNs have the average mobility and data packet rates, based on which the optimal number of access routers in a regional area is derived to be applicable to all MNs in the system.

our idea extends from our work in wireless personal communication systems [2]. We use a client-side proxy to support caching and mobility management in Mobile IPv6. The proxy has three functions: (1) working as a GFA as in regional registration to keep tracking MH's location; (2) acting as a service proxy for services engaged by the MH; (3) allocating a buffer space to store service context information for each MH. The proxy will receive invalidation reports from the server on behalf of the MH. If connected, the proxy will forward the invalidation report to the MH. If disconnected, the proxy will store invalidation reports in the proxy's buffer. Once the MH is reconnected, the MH will get the latest invalidation reports from the proxy. The contributions of the paper are (1) the notion of integrated cache and mobility management scheme to minimize the overall network traffic cost; (2) identifying the optimal proxy setting including the region area size that will minimize the overall network traffic generated; (3) demonstrating the benefit of integrated mobility and cache consistency management in MIPv6 compared with no-proxy and/or no-caching schemes.

The rest of the paper is organized as follows. Section 2 describes the proposed integrated cache and mobility management scheme. Section 3 develops a performance model to analyze the cost incurred. Section 4 compares the proposed scheme with no-proxy and/or no-caching schemes in Mobile IPv6 systems. Finally, Section 5 summarizes the paper and discusses the applicability.

2 Integrated Cache and Mobility Management

When a MH starts in a Mobile IPv6 environment, a client-side proxy is created. We assume that access routers (ARs) are powerful and flexible in future all-IP based wireless networks. Therefore, a client-side proxy can execute on ARs to perform network-layer and application-layer functions on behalf of the client, like programmable agents in wireless IP systems [1, 8].

As illustrated in Figure 1, to support cache management, the client-sider proxy maintains a buffer to store invalidation reports received from the server. The proxy communicates with the server on behalf of the MH. Thus, invalidation reports and data packets sent to the MH from the server will be routed through the proxy, even when the MH is disconnected.

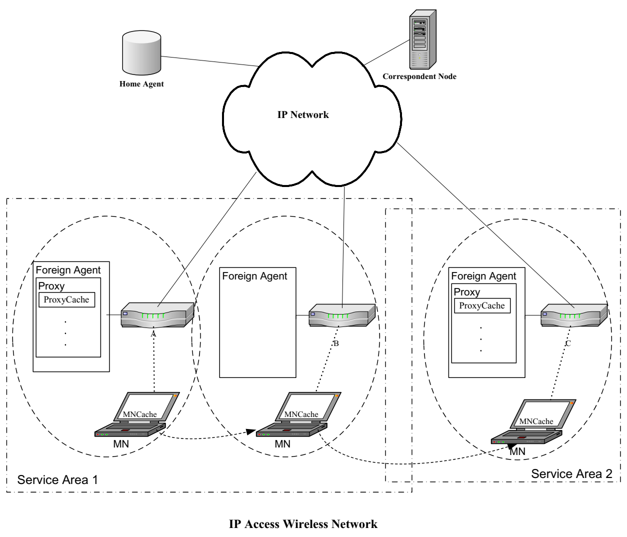

Figure 1: System Infrastructure.

To support mobility management, the client-sider proxy also serves as the Gateway Foreign Agent (GFA) as in the Regional Registration protocol to maintain the location information of the MH. When the MH moves across a subnet boundary within the GFA area, it obtains a new CoA. However, the MH does not need to inform the HA and CNs of the CoA change; instead only the proxy is informed of the CoA change. On the other hand, when the MH moves across a GFA area[1], the proxy also moves to run on the AR of the first subnet in the new GFA area. This incurs the cost of transferring the proxy from one GFA to another GFA area, and also the cost of informing the HA and CNs of the address change of the proxy. The size of the service area is defined as the number of subnet it covers. The optimal size of the service area depends on a MH's runtime mobility and service characteristics to minimize the mobility and service management cost.

To improve query performance, the MH may store frequently used data objects in its cache (called

[1]A GFA area is termed a "service area" in this paper because we consider integrated mobility and service management

MHCache). The MH will use invalidation reports received from the CN (through the proxy) to validate the content of its cache. We consider that the system operates based on the asynchronous *stateless* strategy, that is, when a data object is changed, the CN immediately sends out an invalidation report to those MHs that keep a cached copy. When a service proxy is created to run on an AR, the proxy is allocated with a buffer space (called *ProxyCache*) to cache invalidation reports and possibly application context sensitive information. The proxy acting on behalf of the MH receives invalidation reports from the CN. If the MH is connected, the proxy forwards them to the MH. If the MH is disconnected, the proxy stores them in *ProxyCache*. Once the MH wakes up, it gets invalidation reports from *ProxyCache* to check if its cache content is current. When the proxy moves because the MH moves across a service area, *ProxyCache* moves with the proxy, thus incurring a *context transfer* cost.

A trade-off exists to balance the mobility management cost and the cache consistency management cost. A large service area size means that the proxy will not move often, and the query cost due to cache misses could be high because of the triangular routing path CN-Proxy-MH. The cache invalidation cost is also high because of the larger distance between the proxy and the MH when the service area is large. A small service area means that the proxy moves often. This increases the cost of moving ProxyCache with the proxy and to inform the HA and CNs of address change. Thus, an optimal service area size exists. The proxy determines the *dynamic* optimal service area size at runtime. When the MH moves across a subnet boundary, the proxy will check if the service area is crossed. If yes, a new optimal service area size is determined by executing a computational procedure developed in this paper based on the MH's mobility and service characteristics in the new service area.

3 Performance Analysis

3.1 Model

We develop a performance model based on Stochastic Petri nets to analyze the cache management cost and mobility management cost incurred due to the employment of the integrated mobility and cache management scheme. The objective is to derive an equation from the analytical model to allow the overall network traffic cost incurred to be calculated as a function of the number of subnets covered (K) in a service area, when given a set of parameters as input to characterize a MH's mobility and service behaviors. This computational procedure allows us to determine the optimal service area size (K_{opt}) to be deployed at runtime to minimize the overall network traffic cost due to cache and mobility management.

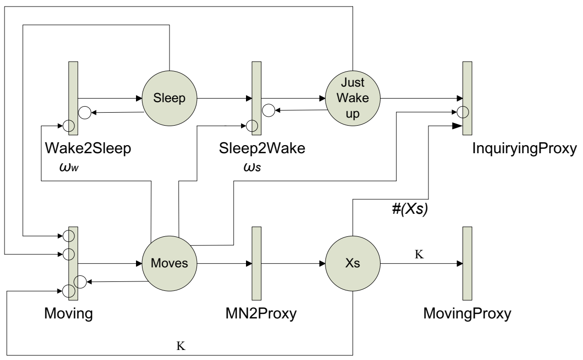

Figure 2: Petri Net Model.

Figure 2 shows the performance model based on Stochastic Petri nets. Let the function Mark(`P`) return the number of tokens in place `P`. The number of tokens accumulated in place `Xs`, that is, Mark(`Xs`), represents the number of subnets crossed by the MH since the MH enters a new service area. We allow it to accumulate to K, at which point we perform a service handoff. We construct this SPN model to describe the behavior of a MH operating under our proposed integrated cache and mobility management scheme. By varying K, the SPN model allows us to compute the cache and mobility management cost as a function of K, thereby allowing the optimal K to be determined.

Below we describe how we construct the SPN model: After moving into a subnet, the MH obtains a new CoA and informs the proxy (that acts as a GFA) of the CoA change. This is modeled by enabling and firing transition `MH2Proxy` while disabling transition `Moving`. After `MH2Proxy` is fired, a token in place `Moves` flows to place `Xs`, representing that a location handoff has been completed and the proxy has been informed of the CoA address change of the MH. The rate at which `MH2Proxy` is fired depends on the number of subnets separating the MH and the proxy.

If the number of tokens in place `Xs` has accumulated to K, a threshold representing the size of a service area, it means that the MH has just moved into a new service area and a service handoff ensues. This is modeled by assigning an function that will enable transition `MovingProxy` after K tokens have been accumulated in place `Xs`. After transition `MovingProxy` is fired, all K tokens are consumed and place `Xs` contains no token, representing the action that the proxy has just moved into a new service area. The rate at which transition `MovingProxy` fires depends on the cost of informing the HA and CNs of the proxy CoA change and the cost of transferring *ProxyCache* to the new proxy location.

After the MH wakes up, it will check with the proxy of the status of its cached data objects. If the proxy is not in the same subnet, then the proxy moves to the subnet that the MH currently resides. The cost involved includes the cost of service context transfer of *ProxyCache* and the signaling cost of informing the HA and CNs of the address change. This is modeled by firing the `inquiryProxy` transition with a transition rate reflecting the cost (see how we do parameterization below). Firing this transition will flush all the tokens in place `Xs` as if a service handoff had happened. This is modeled by a variable input arc from place `Xs` to transition `inquiryProxy`.

3.2 Parameterization

We let $F(K)$ denote a function mapping the number of subnet crossings K to the number of hops. In this paper we adopt the fluid flow model [9] assuming that the average number of hops between two communicating hosts separated by K subnets is equal to $\sqrt{K}$.

Below we describe how we parameterize the SPN model. Transitions `Moving`, `wake2Sleep` and `sleep2Wake` have transition rates of σ, ω_w and ω_s, respectively. Here, σ is the mobility rate; ω_w is the disconnection rate from awake to asleep; ω_s is the reconnection rate from asleep to awake. Transition rates of the three remaining transitions need to be parameterized (i.e., given values to) given basic parameter values. The firing time of transition `MH2Proxy` stands for the communication time of the MH registering with the proxy through the wireless network. This time depends on the number of hops separating the MH and its proxy. Thus, the transition rate of transition `MH2Proxy` is calculated as $1/[\gamma\tau + F(Mark(\mathrm{XS})+1)\times\tau]$ where τ stands for the one-hop communication delay per packet in the wired network and γ is a proportionality constant representing the ratio of the communication delay in the wireless network to the communication delay in the wired network. $F(\mathrm{Mark}(\texttt{Xs})+1)$ returns the number of hops between the current subnet and the proxy separated by $\mathrm{Mark}(\texttt{Xs}) + 1$ subnets. The argument of the $F(x)$ function is added by 1 to satisfy the initial condition that $\mathrm{Mark}(\texttt{Xs})=0$ in which the proxy has just moved into a new service area, so at the first subnet crossing event, the distance between the proxy and the subnet is one subnet apart. Note that this transition rate is state-dependent because the number of tokens in place `Xs` changes dynamically over time.

When transition `MovingProxy` fires, the proxy will move into a service area after invoking a service handoff. The cost involved includes informing the HA and CNs of the CoA address change, and transferring service context information stored in *ProxyCache*. The transition rate of transition `MovingProxy` thus is calculated as $1/[n_{CT}F(K)\tau + (\alpha + N\beta)\tau]$ where n_{CT} is the number of packets required to carry the service context information during a proxy transfer, α is the average distance between the proxy and the HA, N is the number of server applications (or CNs) which the MH engages concurrently, and β is the average distance between the proxy and a CN. The proxy could determine the values of α and β based on statistical data collected on the fly.

When transition `InquiryProxy` fires, the MH will contact the proxy. If the proxy is in the same subnet as the MH resides, the cost is the transmission delay from the AR to the MH, that is, $\gamma\tau$. If the proxy is not in the same subnet, the cost includes the delay for contacting the proxy, moving the proxy to the current AR, and informing the HA and CNs of the proxy's CoA address change. Thus the transition rate of transition *InquiryProxy* is calculated as:

$$\begin{cases} \dfrac{1}{\gamma\tau} & \text{if Mark(Xs)} = 0; \\ \dfrac{1}{\{F(\mathrm{Mark(Xs)}) + n_{CT}F(\mathrm{Mark(Xs)}) + (\alpha + N\beta + \gamma)\}\tau} & \text{if Mark(Xs)} > 0. \end{cases}$$

3.3 Cost Function Derivation

Let C_{query} be the average communication cost to service a query. Let $C_{mobility}$ be the average communication cost to service a location handoff, including one that can trigger a service handoff. Let $C_{invalidation}$ be the average communication cost to forward an invalidation report. Finally let C_{total} be the overall cost incurred *per time unit*. Then, C_{total} is the sum of the product of the respective communication cost multiplied with the rate at which the respective event occurs, that is,

$$C_{total} = \lambda_e \times C_{query} + \sigma_e \times C_{mobility} + \mu_e \times C_{invalidation}$$

where λ_e is the effective data query rate, σ_e is the effective mobility rate, and μ_e is the effective data update rate.

The effective query arrival rate λ_e is the query arrival rate multiplied with the probability of the MH is being awake because queries are issued only when the MH is awake. That is, $\lambda_e = \lambda_Q \times P_{wake}$, where λ_Q is the aggregate query arrival rate, given by $\lambda_Q = \sum_{i=1}^{N_{data}} \lambda_{q,i}$, and P_{wake} is the probability of the MH being in the wake state, given by $P_{wake} = \omega_s/(\omega_s + \omega_w)$.

The effective mobility rate σ_e is the mobility rate multiplied with the probability of the MH being awake again because the MH does not trigger mobility handoff events while it is in sleep model. Thus, $\sigma_e = \sigma \times P_{wake}$.

On the other hand, the effective data update rate μ_e is simply the aggregate data update rate, calculated as:

$$\mu_e = \sum_{i=1}^{N_{data}} \mu_i.$$

where μ_i is the update rate to object i, the expression for μ_e is not multiplied with the probability of the MH being awake because updates to data occur regardless of if MH is in sleep or wakeup mode.

Below we derive C_{query}, $C_{mobility}$ and $C_{invalidation}$. The stochastic model underlying the SPN model is a continuous-time semi-Markov chain[2] with the state representation of (a, b, c, d) where a is the number of tokens in place `Moves`, b is the number of tokens in place `Xs`, c is the number of tokens in place `Sleep`, and d is the number of tokens in place `Just Wake Up`. The distribution of tokens in these four places makes up the states of the system. In general let state i represent a particular state represented by (a, b, c, d). Let P_i be the steady state probability that the system is in state i. P_i's can be obtained by applying numerical analysis methods such as SOR or Gauss Seidel to solve the underlying model.

Let $C_{i,query}$ be the communication cost for answering a query given that the MH is in state i. Then, C_{query} can be calculated as a weighted average of $C_{i,query}$'s as follows:

$$C_{query} = \sum_i (P_i \times C_{i,query}) \qquad (1)$$

Deriving $C_{i,query}$ requires knowledge of $P_{miss,j}$, i.e., the probability of cache miss of data object j, because this cost depends on whether the requested cached data object is up-to-date and can be used to answer a query. A cache miss happens if an update has occurred to object j prior to the query requesting object j arriving at the MH. Inevitably this depends on the relative magnitudes of $\lambda_{q,j}$, μ_j and ω_w and ω_s. The calculation of $P_{miss,j}$ essentially hinges on the competition between the *effective query arrival rate* and the update rate (with respect to time), since the cached data object will be invalidated when an update occurs before a query arrives. Since the MH will not issue queries while it is in sleep mode, the effective query arrival rate for object j is equal to the query arrival rate for data object j multiplied with the probability of being awake, i.e., $P_{wake}\lambda_{q,j}$, or $[\omega_s/(\omega_s + \omega_w)]\lambda_{q,j}$. Thus, $P_{miss,j}$ is calculated as:

$$P_{miss,j} = \frac{\mu_j}{(\mu_j + [\omega_s/(\omega_s + \omega_w)]\lambda_{q,j})}$$

Suppose that a query or a subquery is asking for data object j. If data object j being queried is in the MH's cache and is valid, then there is a cache hit and the query cost is zero. Otherwise, there is a cache miss, and the query cost will include a communication cost between the MH to the proxy, and a cost from the proxy to the CN. Thus, the average query cost for a query asking for data object j when the MH is connected is given by $(\gamma\tau + \beta\tau + F(\text{Mark(Xs)})\tau) \times n_D \times P_{miss,j}$ where n_D is the number of packets to hold a data object.

On the other hand, when the MH just wakes up, i.e., in state "just wake up", the MH will first check with the proxy regarding the cache status when answering a query, and, if the cached data object is not valid, will get a copy from the server. Thus, the cost is given by $\gamma\tau + (\gamma\tau + \beta\tau + F(\text{Mark(Xs)})\tau) \times n_D \times P_{miss,j}$ where the first term is the cost for the MH to get invalidation reports from the proxy (which has moved to local) to check the cache status and the second term is for the cost to get a copy of object j from the CN if there is a miss. Lastly, when the MH is in sleep mode, there is no query cost because no query is issued while the MH is in sleep. Thus, $C_{i,query}$ is calculated as:

$$\begin{cases} 0 & \text{if Mark(Sleep)} \\ \gamma\tau + (\gamma\tau + \beta\tau + F(\text{Mark(Xs)})\tau) n_D P_{miss,j} & \\ & \text{else if Mark(Just Wake Up)} > 0 \\ (\gamma\tau + \beta\tau + F(\text{Mark(Xs)})\tau) \times n_D \times P_{miss,j} & \\ & \text{otherwise.} \end{cases}$$

Let $C_{i,mobility}$ be the communication cost to service a location handoff given that the MH is in state i. $C_{mobility}$ is calculated as a weighted average as:

$$C_{mobility} = \sum_i (P_i \times C_{i,mobility})$$

If in state i, Mark(Xs) $< K$, then the MH will only inform the proxy of the CoA address change. On the other hand, if Mark(Xs) $= K$, then the location handoff also triggers a service handoff. A service handoff will incur a context transfer cost of *ProxyCache* while moving the proxy to the new service area and a communication cost to inform the HA and N CNs (or application servers) of the CoA address change of the proxy. When the MH is in the sleep state, there is no location handoff cost since the MH is disconnected from the system. When the MH just wakes up, it will look for its proxy. If the proxy is in the same subnet as MH currently resides because the MH does not move during sleep, the cost involved is only for contacting the current AR for the proxy location. Otherwise, the proxy is moved to the current subnet, and the cost is that of a service handoff, i.e., a context transfer cost and a cost to inform the HA and N CNs of the CoA address change of the proxy. Therefore, $C_{i,mobility}$ is given by:

[2]If the elapsed time of a timed transition is generally distributed then the underlying model is a semi-Markov chain; if the elapsed time is exponentially distributed, then the underlying model is a Markov chain.

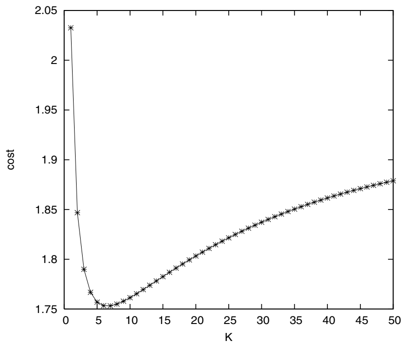

Figure 3: Cost vs. K.

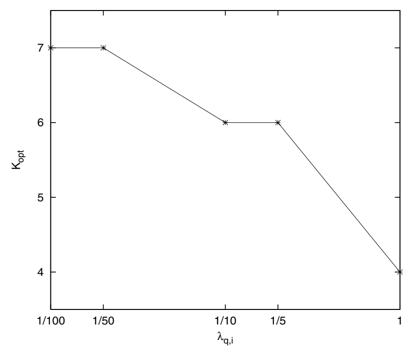

Figure 4: K_{opt} vs. $\lambda_{q,i}$

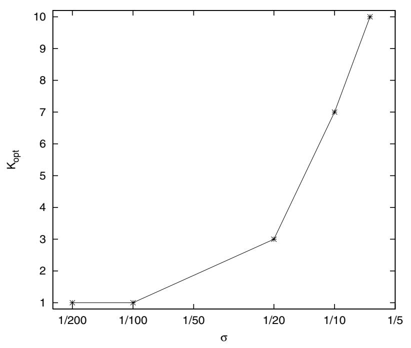

Figure 5: K_{opt} vs. σ.

$$\begin{cases} 0 & \text{if Mark(Sleep)} > 0 \\ \gamma\tau & \text{else if Mark(Just Wake Up)} > 0 \text{ and Mark(Xs)} = 0 \\ \gamma\tau + \alpha\tau + N\beta\tau + F(\text{Mark(Xs)})n_{CT}\tau & \\ \quad & \text{else if Mark(Just Wake Up)} > 0 \text{ and Mark(Xs)} > 0 \\ \gamma\tau + F(\text{Mark(Xs)})\tau & \text{else if Mark(Xs)} < K \\ \gamma\tau + \alpha\tau + N\beta\tau + F(K)n_{CT}\tau & \text{else if Mark(Xs)} = K \end{cases}$$

Lastly, let $C_{i,invalidation}$ be the cost to forward an invalidation report from the proxy to the MH when the MH is in state i. Similarly, the weighted cost per invalidation report is calculated as $C_{invalidation} = \sum_i (P_i \times C_{i,invalidation})$. When an update occurs, the CN sends an invalidation report to the MH. If the MH is in sleep or just wake-up mode, then the invalidation report is buffered in the proxy, so the cost is from the CN to the proxy. Otherwise the cost is from the CN through the proxy to the MH. Thus, $C_{i,invalidation}$ is given by:

$$\begin{cases} \beta\tau & \text{if Mark(Sleep)} > 0 \text{ or Mark(Just Wake Up)} > 0 \\ \beta\tau + F(\text{Mark(Xs)})\tau + \gamma\tau & \text{otherwise} \end{cases}$$

Summarizing above, equations derived in this section allows one to calculate the network traffic incurred per time unit for a MH characterized by a set of parameter values.

4 Numerical Example

In our proposed scheme, a MH and its proxy would apply the above equations in section 3 to calculate C_{total} as a function of K and determine the optimal K representing the optimal *service area* size that will minimize the network signaling cost. Below we present numerical results.

To provide a better sense of the performance improvement of our proposed proxy-based scheme for integrated cache and mobility management, we compare our scheme with three schemes, namely, a *no-proxy no-caching* (NPNC) scheme, a *proxy no-caching* (PNC) scheme, and a *no-proxy caching* (NPC) management scheme. The NPNC scheme essentially is the basic MIPv6 scheme without using a proxy for either mobility or cache management. The PNC scheme is the proxy-based regional registration scheme using a proxy for mobility management. In these two schemes, the MH does not cache data objects. The NPC scheme uses basic MIPv6 for mobility management and cached data objects maintained by the MH for cache management, but there is no proxy being used. Please note that the cost is on a per-sec and per-MH basis so the cumulative cost saved for all MHs over time is significant.

Figure 3 shows that under our integrated scheme there exists an optimal proxy service area size K_{opt} to minimize the overall network traffic cost, when given a set of parameter values characterizing the mobility and service behaviors of the MH in Mobile IP networks.

We observe from Figure 4 that K_{opt} exists for all $\lambda_{q,i}$ values. We see that K_{opt} decreases slowly as $\lambda_{q,i}$ increases. The reason is that as $\lambda_{q,i}$ increases, the query cost increases, and subsequently the MH prefers a small service area size to reduce the query cost.

We observe from Figure 5 that K_{opt} increases as σ increases. The reason is that when the mobility rate is high, the mobility management cost is also high. Therefore, the proxy likes to stay at a large service area to reduce the location handoff cost such that a location handoff will most likely only involve informing the proxy of the location change without incurring a service handoff to migrate the proxy. As a result, when σ increases, K_{opt} increases.

We observe from Figure 6 that K_{opt} decreases as μ_i increases. The reasons is that as μ_i increases, the data in the local cache are more likely to be out-of-date. The MH in this case will more likely to send queries to the server to obtain the latest version of data through the proxy. Also more invalidation reports will be sent from the CN to the MH through the proxy. Therefore, the MH will stay close to the proxy to reduce the triangular CN-proxy-MH communication cost.

Figure 7 compares our proposed PICMM scheme vs. NPNC, PNC and NPC management schemes in

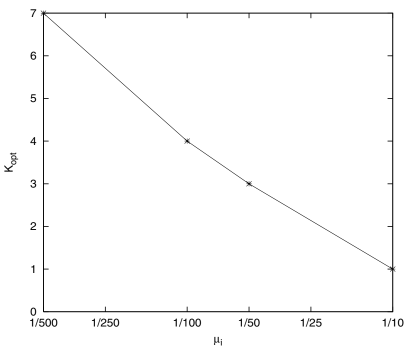

Figure 6: K_{opt} vs. μ_i.

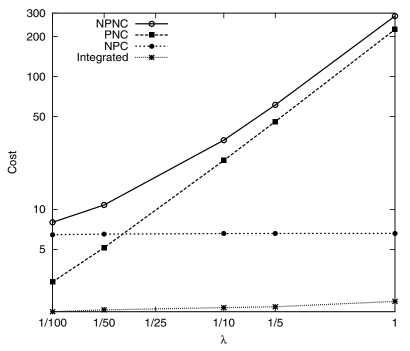

Figure 7: Generated Network Traffic as a Function of $\lambda_{q,i}$.

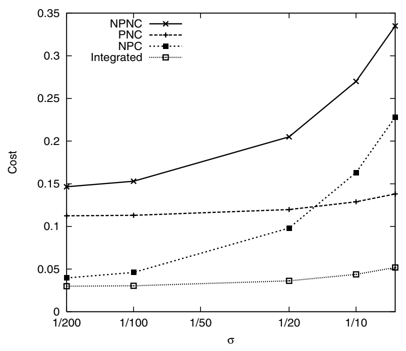

Figure 8: Generated Network Traffic as a Function of σ.

the network traffic cost generated, as a function of $\lambda_{q,i}$ with all other parameters fixed. For the PNC scheme, the cost shown is the minimized network traffic generated at the optimal service area. From Figure 7 we see that caching based schemes (NPC and PICMM) achieve much better performance than non-caching based schemes (NPNC and PNC), especially when $\lambda_{q,i}$ is large. The cost of either NPC or PICMM increases slowly as $\lambda_{q,i}$ increases, while the cost for either PNC or NPNC increases drastically as $\lambda_{q,i}$ increases. The reason is that at high $\lambda_{q,i}$ values, the dominating network traffic is due to sending queries to the server. Caching-based schemes can greatly reduce the magnitude of server-querying network traffic because caching allows some of the queries to be processed by the MH based on up-to-date cached data objects. This trend is true when the update rate to cached data objects (μ_i) is low to modest so that the cost saving due to query processing for sending queries to the server outweighs the extra cost due to triangular routing for receiving invalidation reports and query replies. On the other hand, between the two caching-based schemes, our PICMM scheme performs consistently better than NPC due to the use of a proxy for integrated cache and mobility management for extra cost saving.

Figure 8 compares our proposed PICMM scheme vs. NPNC, PNC and NPC management schemes in the generated network traffic, as a function of σ with all other parameters fixed. Figure 8 shows that our proposed PICMM scheme outperforms all other schemes. Compared with the NPNC scheme, the performance of our scheme is especially pronounced when σ is high. The reason is that PICMM uses a proxy to serve as a GFA to reduce the cost of location handoffs and the benefit is especially pronounced when the mobility rate of the MH is high. Compared with the NPC scheme, PICMM considers integrated cache management and mobility management. Consequently, the best service area determined can minimize the overall network traffic cost better than that by NPC which runs mobility management independently of cache management.

In Figure 9 we compare PICMM vs. other schemes in the generated network traffic, as a function of the cache size. The total cost incurred under PICMM increases as the number of cached data items increases. The main reason is that the invalidation cost increases as the number of cached data items increases. Compared with non-caching based schemes (NPNC and PNC), PICMM achieves much better performance especially when N_{data} is large because caching saves much of the uplink cost for query processing.

In Figure 10 we compare PICMM vs. NPNC, PNC and NPC management schemes in the network traffic generated, as a function of μ_i with all other parameters fixed. Under a non-caching based scheme (NPNC or PNC), the MH will always send queries to the server for processing, since the MH does not cache data objects. As a result, the generated network traffic is insensitive to μ_i. On the other hand, when caching is used as in PICMM or NPC, the generated network traffic becomes sensitive to μ_i because the query traffic to the server depends on if cached data objects are valid. We see that PICMM consistently performs better than NPC. However, when the data update rate is very high, most cached data objects are invalid, so queries will need to be routed to the CN. In this case, there exists a cross-over point in the update rate beyond which PICMM would perform worse than a non-caching scheme because of the CN-proxy-MH triangular cost for routing query inquiries/replies and invalidation reports. In general in cases data are being updated frequently, a caching scheme would not perform better than a non-caching scheme, even with the use of a smart proxy as in PICMM for optimized, integrated cache and mobility management. In this extreme case, the system is better off with the proxy no-caching scheme (PNC).

In Figure 11 we compare PICMM vs. NPC, PNC and NPNC in the generated network traffic, as a function of the sleep ratio ω_w/ω_s. When the sleep ra-

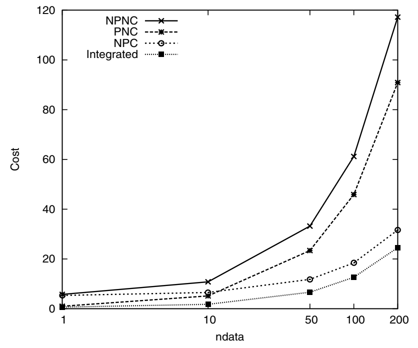

Figure 9: Generated Network Traffic as a Function of Cache Size.

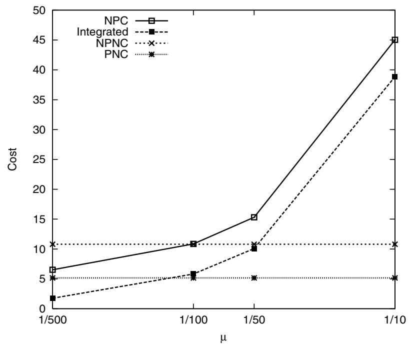

Figure 10: Generated Network Traffic as a Function of μ_i.

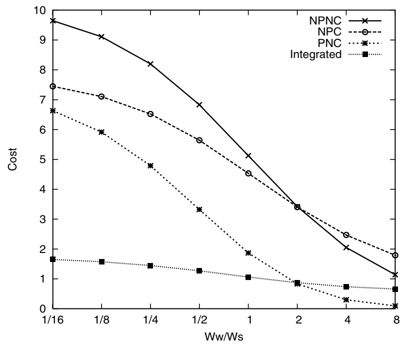

Figure 11: Generated Network Traffic as a Function of ω_w/ω_s.

tio ω_w/ω_s is extremely large, MH is mostly sleeping all the time. The cache is mostly invalid due to long sleep. In this extreme condition, PICMM incurs a higher query cost than non-caching schemes although the cost is small since P_{wake} is small. Again we see that there is a cross-over point in the sleep ratio beyond which PICMM would perform worse than a non-caching scheme such as PNC because of the CN-proxy-MH triangular cost for routing query inquiries/replies and invalidation reports under PICMM. We see that, however, under a reasonable range of sleep ratio, PICMM outperforms all other schemes.

In summary, we observe that our proposed scheme outperforms NPC, PNC and NPNC in terms of the network traffic cost generated over a wide range of parameter values, the effect of which is especially pronounced when the data update rate is low, the mobility rate is high, the query rate is high, the cache size is large, and/or the sleep ratio is low. PICMM greatly reduces the network traffic of mobile query-based applications, except under extreme conditions in which data update rates are exceptionally high and the MH disconnects more than 2/3 of the time during the execution of mobile applications. These extreme conditions, however, rarely happen in practical mobile applications.

5 Applicability and Conclusion

To apply the analysis results presented in the paper, one can execute the computational procedure at static time to determine optimal K_{opt} over a possible range of parameter values for key parameters such as α, β, n_{CT}, N_{data}, $\lambda_{q,i}$, σ and μ_i. Then at runtime a MH can perform a simple look-up operation to determine K_{opt} based on data collected at runtime. This allows each MH to dynamically determine the best service area size to minimize the overall network traffic generated. The performance gain is in the amount of network traffic communication cost saved per time unit per user, so the cost saving due to a proper selection of the best service area dynamically will have significant impacts since the cumulative effect for all mobile users over a long time period would be significant.

In the future, we plan to investigate applications to which the concept of integrated service and mobility management could apply and assess the outcome. We also plan to investigate conditions under which it will be more beneficial for the CN to send copies of objects instead of invalidation reports to MH for cache management.

References

[1] H. de Meer, A. Corte, A. Puliafito, and O. Tomarchio. Programmable agents for flexible qos management in IP networks. *IEEE Transactions on Selected Areas in Communications*, 18(2):256–267, 2000.

[2] B. Gu and I. R. Chen. Performance analysis of location-aware mobile service proxies for reducing network cost in personal communication systems. *ACM Mobile Networks and Applications*, 10(4):453–463, 2005.

[3] E. Gustafsson, A. Jonsson, and C. Perkins. *Mobile IPv4 Regional Registration*. http://www.ietf.org/internet-drafts/draft-ietf-mip4-reg-tun nel-02.txt, IETF, Work in Progress, 2006.

[4] D. M. Huizinga and P. Mann. Disconnected operation for heterogeneous servers. In *ACM symposium on Applied Computing*, February 1996.

[5] S. Mtika and F. Takawira. Mobile IPv6 regional mobility management. In *ACM 4th international symposium on Information and communication technologies*, pages 93–98, Cape Town, South Africa, 2005.

[6] E. Pitoura and G. Samaras. *Data Management for Mobile Computing*. Kluwer Academic Publishers, 1998.

[7] H. Soliman, C. Castelluccia, K. El-Malki, and L. Bellier. *Hierarchical Mobile IPv6 Mobility Management*. IETF, Work in Progress, 2005.

[8] P. Zerfos, G. Zhong, J. Cheng, H. Luo, S. Lu, and J. J.-R. Li. Dirac: a software-based wireless router system. In *9th annual international conference on Mobile computing and networking*, pages 230–244, 2003.

[9] X. Zhang, J. Castellanos, and A. Campbell. P-MIP: Paging extensions for Mobile IP. *Mobile Networks and Applications*, 7(2):127–141, Mar. 2002.