Caching in hierarchical user location databases for PCS

*Ravi Jain**
Applied Research
Bellcore

Farooq Anjum
Dept. Of Elect. Eng. and ISR
Univ. Of Maryland, College Park

Abstract. Mobility management architectures for future generations of PCS systems have been proposed where multiple location databases, organized in a tree, are used to cope with the large number of users. We have previously proposed the use of caching as an auxiliary strategy for reducing the network impacts of locating mobile users in a tree-structured architecture. In this paper we quantify the costs and benefits of caching for one particular variation of a caching strategy, called *eager caching*. Unlike our previous work where we only considered the user's calling and mobility behavior in terms of the aggregated Regional Call-to-Mobility Ratio (RCMR), in this paper the performance evaluation is done in terms of the Local Call-to-Mobility Ratio (LCMR) of the user. We show that under certain assumptions an eager caching strategy for users whose LCMR exceeds 5 can result in substantial reductions in total network cost as well as call setup time.

1 Introduction

Studies have shown that, with predicted levels of Personal Communications Services (PCS) users, there will be significant loads upon network databases and infrastructure in order to locate and deliver calls to mobile users [6, 8]. Thus a distributed database architecture has been proposed in which the databases are organized in a tree [7, 9, 1, 2], and is under study for the European third-generation mobile system called the Universal Mobile Telecommunication System (UMTS) [2].

A tree-structured location database architecture helps to reduce network signalling traffic when most calls received by users and most registration area crossings are geographically localized. However, the total number of databases queried, and hence call setup time, can potentially increase. We have proposed that the judicious use of caching in tree-structured architectures can help alleviate this problem [4].

In tree location strategies [9, 1], as in most other PCS user location strategies, the same user location procedure is invoked for every call to a mobile user. For the tree strategy, in general the procedure involves signaling messages traversing the shortest path in the tree between two leaves, where the leaves are either the locations of the caller and the called party (in the case of call delivery) or the old and new locations of the mobile user (in the case of registration.) The key observation is that, in many cases, it should be possible to re-use the information about the user's location from the previous call to that user. In a tree architecture, this information essentially consists of a *bypass pointer* kept at a database along the path that allows the signaling messages to bypass portions of the path, thus saving communications and database costs. Clearly, this information will be useful for those users who receive calls frequently relative to the rate at which they cross registration areas, i.e., users with a high *Call-to-Mobility Ratio* (CMR).

In a previous paper [3] we have investigated one particular variation of caching, called *eager caching*. The performance of caching in tree architectures depends upon where in the tree bypass pointers are maintained. We defined the user's Regional Call-to-Mobility Ratio (RCMR) as the average rate of calls to the user from the subtree where the pointer was maintained, to the average rate at which the user moved out of the subtree rooted at the node to which the pointer is directed. (This definition is illustrated below.) We developed a simplified "first-cut" analytical model and quantified the costs and benefits of caching in terms of the user's RCMR. However, it is more convenient to calculate as well as to think in terms of the user's Local CMR (LCMR), which is the rate at which the user receives calls from a particular registration area to the rate at which it moves between registration areas. In the present paper we develop an analytical model of the costs and benefits of caching in terms of the user's LCMR. With this model it can be seen that, provided the databases involved can handle the query load offered, the maximum benefits of caching are obtained by maintaining bypass pointers at the lowest level of the tree (unless the user's LCMR is extremely low.)

We define the network model and the basic user location strategies used in tree-structured architectures

*Address for correspondence: Bellcore, 445 South St, Morristown, NJ 07960. Phone: (973)-829-3178. Fax: (973)-829-2645. Email: rjain@bellcore.com

in sec. 2. We discuss caching for tree architectures in sec. 3 and describe eager caching; this material is repeated from our previous paper [3] in order to keep the present paper self-contained. In sec. 4 we present the analysis the costs and benefits of this strategy, and in sec. 5 show the results of evaluating the model for different scenarios. We end with brief remarks.

2 System model and basic strategy

The basic system model and user location strategy is the same as presented earlier [3]. We summarize it here for completeness; see [3] for details. Databases are used to store information about the location of PCS users, and the databases are interconnected by the links of the signalling network, e.g. a Common Channel Signalling (CCS) network, in the form of a tree [9, 1, 2]. (See Fig. 1.) The database at each leaf of the tree serves a single Registration Area (RA). Thus, in Fig. 1, databases i and j serve RAs i and RA j respectively, and maintain information about PCS users registered in those RAs. The database which is the *least common ancestor* of i and j is denoted $LCA(i,j)$, and maintains information about users registered at all the leaves in its subtree.

The strategy used to locate mobile users consists of a $BasicFIND$ operation, when a subscriber tries to place a call to a PCS user, and a $BasicMOVE$ operation when a PCS user crosses an RA boundary.

$BasicMOVE$ consists of the following steps (see Fig. 1). The PCS user moves from the old RA, i, to RA j and registers at database j. A message is sent by database j to its parent (Message 1 in Fig. 1), which is queried to determine if a record for the PCS user exists there. If not, a record pointing to database j is created at the parent database. The message propagates up the tree (Message 2) until a database record is found for the user. This will occur at the database $LCA(i,j)$, and will consist of a pointer to a child database. The message is propagated down the tree following the downward chain of pointers (Messages 3 and 4) until the leaf database serving the user's old RA i is reached. The user is deregistered at the old database, i. An acknowledgement message is propagated up the tree (Messages 5 and 6), and deletes the downward pointers in the databases along its path, until it reaches $LCA(i,j)$. The acknowledgement message is propagated down the tree (Messages 7 and 8) until it reaches the new database, j.

$BasicFIND$ consists of the following steps. A caller at RA i places a call to a PCS user currently located at RA j. Since database i does not contain registration information for the called party, a message is propagated up the tree until a database which does

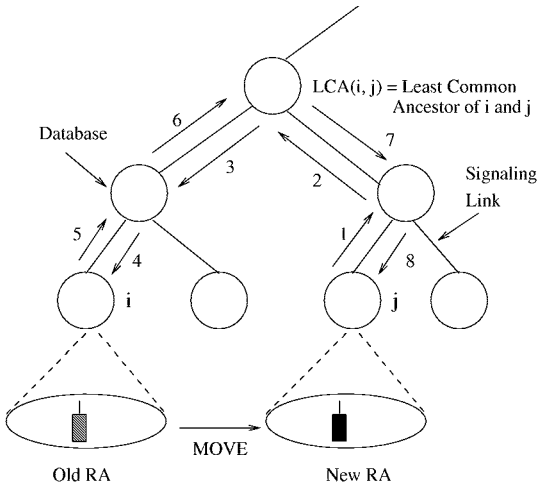

Figure 1: Steps in a $BasicMOVE$ operation

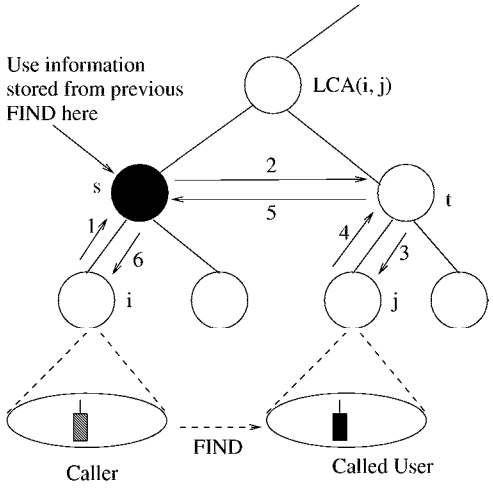

Figure 2: Steps in a $CacheFIND$ operation

contain information is found; this will be $LCA(i,j)$. A message is propagated down the tree from $LCA(i,j)$, following pointers until the leaf database j is reached. Database j returns the information required to set up the call. This return message propagates up the tree until $LCA(i,j)$, after which it propagates down the tree until it returns to the caller's database i.

3 Caching user location

We describe one variation of caching, which we call *eager caching*; see [3] for details. The $BasicFIND$ operation is modified to become the $CacheFIND$ operation (see Fig. 2). When a caller from RA i places a call to a PCS user at RA j, a message propagates up the tree from i to $LCA(i,j)$ and then down the tree to j, and an acknowledgement message returns from j to

i. During the return path, however, a pair of *bypass pointers* is created. A *forward* bypass pointer is an entry at an ancestor of i, say database s, and points to an ancestor of j, say t; the *reverse* bypass pointer is from t to s. (Note that s and t can be at different levels of the tree, although neither can be an ancestor of the other.) During the next call to the same user from RA i, a message traverses up the tree until database s is reached (Message 1 in Fig. 2), and the forward bypass pointer to t is detected. The message travels to database t (Message 2), either via $LCA(i,j)$ or via a shorter route, if it is available in the underlying CCS network. In either case, for the example in Fig. 2, the database at $LCA(i,j)$ need not be consulted. Similarly, the acknowledgement message from the called party's database j will use the reverse bypass pointer (Message 5).

The $EagerMOVE$ operation is as follows. Suppose there exist a forward bypass pointer from database s to t, and a reverse bypass from t to s. When the user moves from RA j to a new RA k, a registration/deregistration message propagates from database k up the tree, via $LCA(j,k)$, to database j, as for a $BasicMOVE$. Several possibilities arise:

1. No bypass pointer is encountered. In that case, s and t are ancestors of $LCA(j,k)$, and their information is valid. No special action required.
2. A reverse bypass pointer is found during the upward traversal of the registration message from k to $LCA(j,k)$. No special action required.
3. A forward bypass pointer is found during the upward traversal, or a (reverse or forward) bypass pointer is found during the downward traversal of the deregistration message (from $LCA(j,k)$ to j.) In that case pointers at both s and t are invalid, and their entries are deleted.

4 Costs and benefits of eager caching

We present an analysis of eager caching using some simplifying assumptions. We consider communication and database processing as our basic measures of cost. Let the cost of traversing any communication link be C, the cost of reading a database be R and updating a database be U. We consider two scenarios: where the underlying CCS network is a tree and provides the only way for messages to be sent from one database to another, or where shortcuts between interior nodes are possible.

If caching is used, the first $CacheFIND$ results in a pair of bypass pointers being created, i.e., an increased cost of $2U$ over $BasicFIND$ (Fig. 2.) Let the path between s and $LCA(s,t)$ be of length b hops and that between t and $LCA(s,t)$ be b' hops. Then subsequent $CacheFIND$ messages enjoy a savings of $2(b'+b-1)R$ over $BasicFIND$ (if the CCS network is a tree) or $2(b'+b-1)R - 2(b'+b)C$ (if shortcuts are available). When the PCS user does move out of the subtree rooted at t, there is an increased cost of deleting bypass pointers of $2U + (b'+b)C$.

Let $p_r(s,t)$ denote the average number of calls from the subtree rooted at s to the PCS user while it is in the subtree rooted at t. We refer to this quantity as the *Regional Call-to-Mobility Ratio (RCMR)* for a user with respect to tree nodes s and t [3]. In this paper we will be concerned with p, the *Local Call-to-Mobility Ratio* (LCMR), which is the average number of calls received by the user from a particular RA (i.e., leaf) while it is located within an RA. The LCMR is just the RCMR for the special case where s and t are constrained to be leaves.

Let the cost of a $MOVE$ operation be denoted by M, the cost of a $BasicFIND$ by F, the cost of a $CacheFIND$ by F' and the cost of a $CacheMOVE$ by M', . Consider a $FIND$ operation where the caller is at RA i, the called party is at RA j and $LCA(i,j)$ is at r levels of the tree above i and j.

When a $BasicMOVE$ occurs, and the user moves from RA j to RA k, the cost depends upon how far apart the two RAs are; suppose that $LCA(j,k)$ is at q levels above j and k. For simplicity, in this example we assume that bypass pointers are always set up between the same levels of the tree, i.e, $b' = b$. Let $T(t)$ denote the subtree rooted at node t.

Let the cost of the basic strategy be denoted by C_B and the cost of the caching strategy by C_C.

We will now consider two measures of performance in order to evaluate the benefits of caching. The ratio $\frac{C_C}{C_B}$ denotes the relative total network cost of caching versus the basic strategy, and the ratio $\frac{F'}{F}$ denotes the call setup time using caching versus the basic strategy. From the considerations above, we can show (assuming no shortcuts are available in the underlying signaling network)

$$F = 2(2b+1)R + 4rC \quad (1)$$

$$M(q) = 2(2q+1)R + (2q+1)U + 4qC \quad (2)$$

$$F' = \begin{cases} F + 2U & \text{first FIND} \\ F - 2(2b-1)R & \text{subsequent FIND} \end{cases} \quad (3)$$

When communication shortcuts are available the same equations apply except $F' = F - 2(2b-1)R - 4bC$ for subsequent FINDs.

Let γ denote the probability that the mobile leaves (i.e., moves out of) $T(t)$ between two consecutive call

arrivals. Then,

$$\gamma = \sum_i \Pr[\text{mobile leaves } T(t) \text{ and there are } i \text{ moves between calls}] \quad (4)$$

$$= \sum_i \gamma(t,i) = \sum_i \alpha(i)\beta(t,i) \quad (5)$$

where $\alpha(i)$ is the probability of i moves between two consecutive calls and $\beta(t,i)$ is the conditional probability of the mobile leaving $T(t)$ given that there are i moves between two calls. Then, if there are no shortcuts,

$$F' = (F+2U)\gamma + (F-2(2b-1)R)(1-\gamma) \quad (6)$$

$$M' = \sum_{i=0}^{\infty} M'(t,i)\alpha(i) \quad (7)$$

where $M'(t,i)$ represents the cost of i $CacheMOVE$s between two consecutive calls given a bypass pointer directed at t. If there are shortcuts available, the equations above remain the same except the factor in the second term of Eq. 6 becomes $(F-2(2b-1)R-4bC)$.

It can be shown [10, 5] that if call arrivals are a Poisson process and the residence time of a user in an RA is exponentially distributed, then

$$\alpha(i) = \frac{p}{(1+p)^{i+1}} \quad (8)$$

4.1 Mobility model

We now define a mobility model in order to derive $M'(t,i)$ and $\beta(t,i)$. We assume that the layout of the tree is such that on average a user's movements among the RAs is uniformly distributed. For simplicity, we assume that the tree is a complete binary tree. Let the maximum height of the entire tree be H. Let the average cost of a basic move be denoted $\overline{M}$, then

$$\overline{M} = \sum_{q=1}^{H} M(q)\frac{2^{H-q}}{2^H-1} \quad (9)$$

$$M'(t,i) = \left[(\overline{M}+2U+2bC)\gamma(t,i)\right] i + \left[\overline{M}(1-\gamma(t,i))\right] i \quad (10)$$

We now obtain a recurrence relation to calculate $\beta(t,i)$, the probability of leaving $T(t)$ given that there are i moves between two calls. For ease of reference let the leaves of $T(t)$ be numbered from 1 to 2^h, where h is the height of $T(t)$. Then

$$\beta(t,i) = \sum_{j=1}^{2^h} Pr[\text{mobile starts in } j]\beta'(t,i,j) \quad (11)$$

where $\beta'(t,i,j)$ is the probability the mobile leaves $T(t)$ in i steps given that it starts in leaf node j. We also let u be the probability of moving to the right and v the probability of moving to the left. Conditioning on the first step we obtain the difference equation

$$\beta'(t,i,j) = \beta'(t,i-1,j+1)u + \beta'(t,i-1,j-1)v, \quad 2 \le j \le 2^h-1, \quad 1 < i \quad (12)$$

$$\beta'(t,i,1) = \beta'(t,i-1,2)u, \quad 1 < i \quad (13)$$

$$\beta'(t,i,2^h) = \beta'(t,i-1,2^h-1)v, \quad 1 < i \quad (14)$$

We have the boundary conditions as

$$\beta'(t,1,1) = v, \quad \beta'(t,1,2^h) = u$$
$$\beta'(t,1,j) = 0, \quad 1 < j < 2^h \quad (15)$$

Finally, we also assume that the probability that the mobile starts in leaf j is $1/2^h$, i.e., that the mobile is equally likely to be in any of the leaf nodes when we start studying the system.

5 Model evaluation

The equations can be used to estimate the value of caching for different types of network costs and user calling and mobility patterns. We have considered two scenarios, where shortcuts in the signaling network are not present and where they are present.

In the first scenario, we have focused on the impact of database costs and set $C = 0$ so as to not mix quantities of different units. As an example, we have plotted $\frac{C_C}{C_B}$ for the case where we assume that database updates are twice as costly as database queries, i.e., $U = 2$, $R = 1$. We assume that the height of the complete tree, $L = 5$. In Fig. 3 we show how the relative costs with the caching strategy vary with LCMR for the bypass pointer placed at different heights. The ratio $\frac{C_C}{C_B}$ decreases as LCMR increases, as expected. It can be seen that for LCMR > 5, caching produces net cost reduction in this example, and with bypass pointers at the first interior node (a height of 1), 45%-60% reduction in costs can be obtained.

Similarly, Fig. 4 plots $\frac{F'}{F}$ versus LCMR under the same assumptions, for various values of the height of the bypass pointer. Again as expected, the benefits of caching increase with LCMR, and for LCMR > 5 caching can result in up to 55% reductions in call setup time in this example.

Finally, we consider the second scenario where shortcuts are available in the network, and we focus on communication costs only, ignoring database costs. Fig. 5 plots C_C/C_B for the case where $C = 1$, $U = R = 0$, and $L = 5$. We see that substantial savings are possible in communications cost.

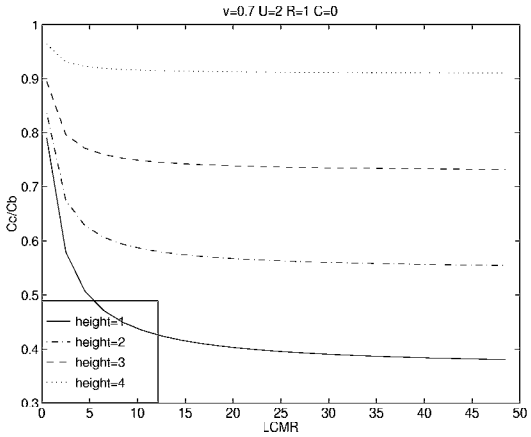

Figure 3: Relative total cost of caching and the basic strategy,for $C = 0, U = 2, R = 1, L = 5$

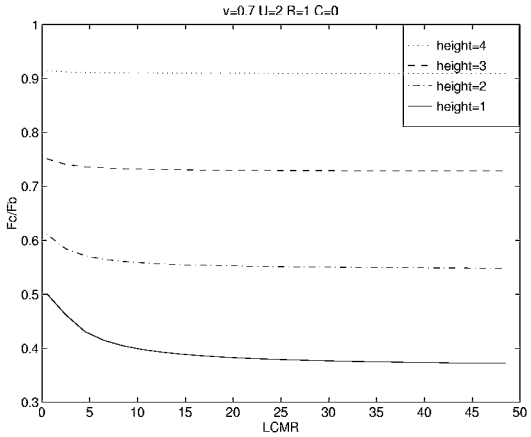

Figure 4: Relative call setup time of caching and the basic strategy, for $C = 0, U = 2, R = 1, L = 5$

6 Concluding remarks

We have developed an analytical model for investigating the costs and benefits of caching in tree-structured database architectures for PCS mobility management. Caching is shown to have substantial benefit, particularly as the user's LCMR increases. In future work we will address algorithms to estimate LCMR dynamically as well as mobility models that exhibit locality.

References

[1] V. Ananthram, M. L. Honig, U. Madhow, and V. K. Wei. Optimization of a database hierarchy for mobility tracking in a personal communications network. In *Proc. Performance '93*, 1993.

[2] C. Eynard, M. Lenti, A. Lombardo, O. Marengo, and S. Palazzo. A methodology for the perfomance evaluation of data query strategies in Universal Mobile Telecommunication Systems (UMTS). *IEEE J. Sel. Areas Comm.*, pages 893–907, Jun. 1995.

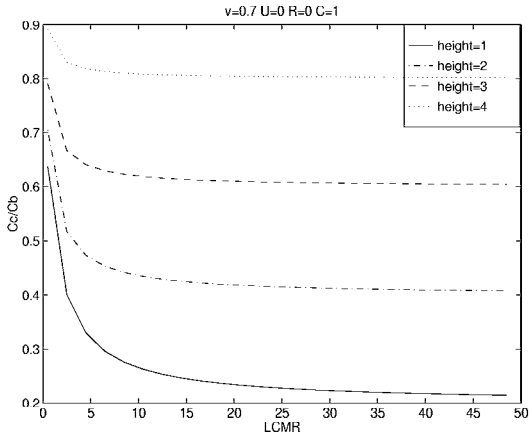

Figure 5: Relative total cost of caching and the basic strategy,for $C = 1, U = R = 0, L = 5$

[3] R. Jain. Locating PCS subscribers using a system of hierarchical user location databases. In *Proc. IEEE Intl. Conf. Comm.*, 1996.

[4] Ravi Jain. Reducing traffic impacts of PCS using hierarchical user location databases. In *Intl. Teletraffic Conf. Mini-Seminar on Mobility*, Montebello, Canada, Oct. 1994.

[5] Ravi Jain and Yi-Bing Lin. An auxiliary user location strategy employing forwarding pointers to reduce network impacts of PCS. *ACM Journal on Wireless Info. Networks*, 1995.

[6] C. N. Lo, R. S. Wolff, and R. C. Bernhardt. An estimate of network database transaction volume to support personal communications services. In *Proc. Intl. Conf. Univ. Pers. Comm.*, 1992.

[7] A. D. Malyan, L. J. Ng, V. C. M. Leung, and R. W. Donaldson. Network architecture and signalling for wireless personal communications. *IEEE J. Sel. Areas Comm.*, pages 830–840, Aug. 1993.

[8] K. Meier-Hellstern and E. Alonso. The use of SS7 and GSM to support high density personal communications. In *Proc. IEEE Intl. Conf. Comm.*, 1992.

[9] J. Z. Wang. A fully distributed location registration strategy for universal personal communications. *IEEE J. Sel. Areas Comm.*, pages 850–860, Aug. 1993.

[10] Y.-B. Lin, S. Mohan, and A. Noerpel. Queueing Priority Channel Assignment Strategies for Handoff and Initial Access for a PCS Network. *IEEE Trans. Veh. Technol.*, 43(3):704–712, 1994.