A Frequency-Based Find Algorithm in Mobile Wireless Computing Systems

Seung-yun Kim and Waleed W. Smari

Department of Electrical Computer Engineering
University of Dayton
300 College Park
Dayton, OH USA 45469

Abstract. The rapid growth in mobile and wireless computing technology continues to present new challenges. Mobile users access information and services independent of their location through wireless and wired networks. In a mobile computing environment, a location management strategy is employed whenever users move from one place to another. In order to track a mobile user (also referred to as a 'Mobile Host', MH), the system must store information about the host's current location and report its new locations to a home base station (referred to as a 'Mobile Base Station', MBS). Several techniques have been proposed in order to manage optimally the location of mobile hosts. In this paper, we introduce the Frequency-Based Find Algorithm (FBFind) with local storages. The basic objective behind using the FBFind is to reduce the Signaling System No. 7 (SS7) message traffic [10] and the database access traffic by maintaining a local storage, which is referred to as the Location Information Library (LIL). We evaluate this algorithm and identify its optimal conditions as well as limitations. Lastly, we compare the algorithm with the caching algorithm proposed by Jain et al [6].
Keywords: Frequency-based Find algorithm; Location management; Mobile/wireless network; Caching algorithm

1. Introduction

Mobile and wireless computing technology continue to witness steep growth. As many of its problems are solved, new issues arise and some existing ones persist. One such challenge is the efficient and optimal management of mobile units in such a system. There are many ways of managing the users' location problem. We divide these into two main basic techniques; look up techniques, which focus on reducing the cost of looking up a user and adjust the update process accordingly, and update techniques, which focus on reducing the cost of updating.

In a mobile and wireless computing environment, mobile hosts may relocate from one cell to another. In order to keep track of mobile hosts, the system must record or know the information about mobile hosts' current location. Elnahas and Adley [3] proposed a scheme to define when an update and lookup occur in a system. Specifically, an update occurs when a mobile host sends a message to update its stored location. A lookup occurs when it is required to locate a user each time a call is placed to that user or a message is sent to him. The location updates and lookups are evaluated in terms of the number of messages sent, the size of messages, the distance the message needs to travel, the bandwidth consumed, the processing overhead and the delay incurred in answering locations queries. A good location management scheme should attempt to optimize these parameters while achieving the seamless mobility. Seamless mobility [1] requires that when connections are broken or established, the process of switching between cells should be done transparently. Pierre [12] discusses the mobility issue as one of three major challenges in mobile wireless computing environments.

Tracking mobile hosts and having efficient routing are two basic functions of a mobile computing system. The system needs to be updated and provided information about the mobile hosts' location. Usually, a location area structure has several cells in it, and a mobile computing environment structure consists of several location areas which may overlap, as shown in Figure 1. In order to reduce the look up and update costs, there have been many proposed techniques. The modified group method [15] yields some improvements over the conventional methods in reducing costs. Moreover, it offers enhancements in the utilization of network resources for mobile hosts that move frequently. Traditionally, an update occurs when a mobile host moves from one cell to another. However, Weng's algorithm would require an update only when a MH moves.

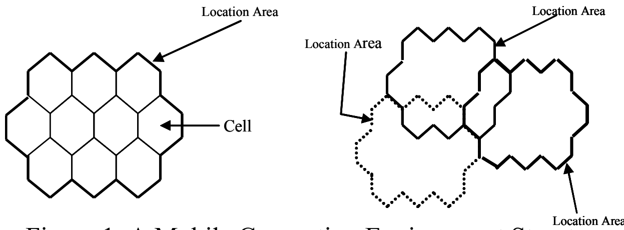


Figure 1. A Mobile Computing Environment Structure

While the above attempts have contributed to reducing the update costs, they do not address detailed costs and other updating issues. The intent of this paper is to discuss the details of all cost functions and to present an algorithm which reduces the costs for locating a Mobile Host in mobile and wireless network environments by using local storage.

This paper is organized as follows. In Section 2, we introduce some basic information on wireless network architectures and a model to help understand the new algorithm. Section 3 presents the FBFind algorithm with all calculations and relationships among model parameters. We evaluate the proposed algorithm in Section 4. In Section 5, we compare the proposed algorithm with another location management algorithm, namely, the caching algorithm and run some experiments. Finally, Section 6 includes some final thoughts and concludes the paper.

2. System Description and Model

Most mobile wireless networks use a cellular architecture to achieve bandwidth efficiency [14]. Over the past decade, several architectural models of mobile and wireless computing systems have been proposed [5, 9].We use the model depicted in Figure 2. A general mobile computing system consists of mobile hosts (MHs) that interact with a static network through fixed hosts, known as mobile base stations (MBSs). A mobile host communicates with one mobile base station at any given time. MBS is responsible for forwarding data between the mobile host and the static network. As a result of mobility, a mobile host may cross the boundary between two cells while being active. Thus, the task of forwarding data between the static network and the mobile host must be transferred to the new cell's mobile base station [8]. To assist in this process, MBSs are augmented with a wireless interface, and function as a gateway for communication between the wireless network and the static network.

The connection between an MH and an MBS is accomplished via wireless communication, while the MBS is wired to the fixed network. A mobile host can communicate with a mobile base station within a limited region around it. This region is referred to as a mobile base station's cell, as shown in Figure 2. Cells can have different sizes. The average size of a cell is about 1 to 2 miles in diameter. A mobile user within a cell receives calls via wireless access. Each cell is primarily served by one MBS, and the MBS serves one cell. In general, calls may deliver voice, data, text, or video information. The location of a user is associated with the registration area (RA) in which the user is located. The base station locates a user and delivers calls to and from that user. It is also connected to a Local Signaling Transfer Point (LSTP) via wired link. Each LSTP has two MBSs since we are considering binary tree architecture from the RSTP level to the MBSs. Likewise, a pair of LSTPs are connected to a Regional Signaling Transfer Point (RSTP). However, several RSTPs may be connected together to a Public Switched Telephone Network (PSTN), which acts as a root of this hierarchical model. Clearly, the system is modeled using a hierarchical location management method [13]. The complete system is illustrated in Figure 2.

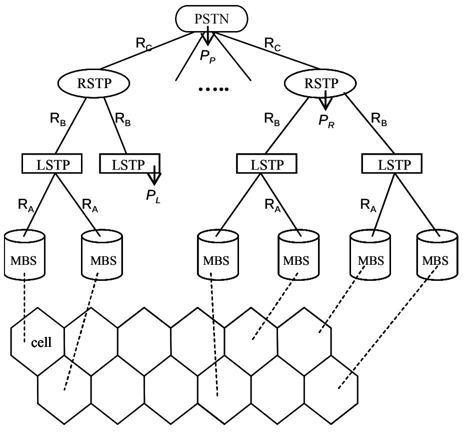


Figure 2. A Mobile Computing System Architecture

2.1. Model Parameters

In the model laid out above, we define six different parameters. Table 1 summarizes the definition of these parameters.

Table 1. Parameter Descriptions

R_A :	Cost of transmitting a location request or response message between links MBS and LSTP
R_B :	Cost of transmitting a location request or response message between links LSTP and RSTP
R_C :	Cost of transmitting a location request or response message between links RSTP and PSTN
P_L :	Cost of routing processes for location request message by LSTP
P_R :	Cost of routing processes for location request message by RSTP
P_P :	Cost of routing processes for location request message by PSTN

We know R_A, R_B, R_C, P_P, P_R and P_L are cost functions, and they are proportional to distance. So, let us assume R_A, R_B, R_C, P_P, P_R and P_L are physical distances between parameters. For simplicity, let us also assume that the distances between MBS and LSTP, and between LSTP and RSTP are the same, i.e., $R_A = R_B = R$, and $P_R = P_L = P$ because we have binary tree up to RSTP. This means that the cost of transmitting a location request or response message between links, and the cost of routing processes for location request message are the same up to the RSTP level. With these assumptions, let us redraw Figure 2 as shown in Figure 3(a). In this figure, we represent the top view of the system architecture model. If we say the distance between MBS and LSTP, and LSTP and RSTP are the same, then we can have the diagram shown in Figure 3(b) to find the relationship between RSTP and PSTN.

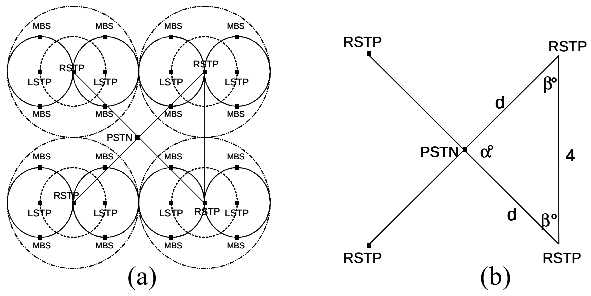


Figure 3. Distances between MBS and LSTP, LSTP and RSTP, and RSTP and PSTN.

The triangle illustrates the relationships between the RSTPs and a PSTN and an RSTP. If we let the distances between an MBS and an LSTP, and an LSTP and an RSTP equals to 1 unit, also for simplicity, then the distance from an RSTN to an RSTN will always be 4 units. Now, let us consider the angles. The angle α can be expressed as $\alpha^\circ = \frac{360^\circ}{t}$, where *t* is the total number of RSTPs. The angle β can be found from the triangle since $\alpha + \beta + \beta = 180^\circ = \frac{360^\circ}{t} + 2\beta$. Solving for β, we get $\beta^\circ = 90^\circ - \frac{360^\circ}{2t}$.

To find the distance *d* between an RSTN and a PSTN, we employ the sine law: $\frac{\sin(\frac{360^\circ}{t})}{4} = \frac{\sin(90^\circ - \frac{360^\circ}{2t})}{d}$.

Solving for d, we obtain $d = \frac{4\cos(\frac{180^\circ}{t})}{\sin(\frac{360^\circ}{t})}$.

If we have *t* RSTPs, we may express the cost parameters R_c and P_p as follows.

$R_C = [4\cos(180°/t) / \sin(360°/t)]R = d{\cdot}R$ ……….. (1)

$P_P = [4\cos(180°/t) / \sin(360°/t)]P = d{\cdot}P$ …..…….. (2)

where R is the cost of transmitting a location request or response message between links, from MBS to LSTP or from LSTP to RSTP, and P is the cost of routing processes for location request message by LSTP or RSTP or PSTN, as was assumed before. Hence, all parameters are now defined.

2.2. User Characterization Using Local Call-to-Mobility Ratio (LCMR)

Classes of users are characterized by their call-to-mobility ratio (CMR) [7]. The call-to-mobility ratio of a user is the average number of calls to a user per unit time, divided by the average number of times the user changes registration areas per unit time. We also define a local CMR (LCMR), which is the average number of calls to a user from a given originating signal transfer point per unit time. The LCMR can be used to relate the hit ratio to users' calling and mobility patterns directly. To do so, we need to make some assumptions about the distributions of the user's calls and moves. Let the call arrivals from an MBS to a user be a Poisson process with arrival rate λ, and the time that the user resides in an RA be $1/\mu$. Then, LCMR can be expressed as: LCMR = λ / μ.

Since we are dealing with non-negative random variables, it is convenient to associate with a probability density function, f(s), its Laplace transform, $f^*(s)$. Let the residual time of a user at a registration area be a random variable with a general density function f_m. Then, its Laplace transform is given by $f_m^*(s) = \int_{t=0}^{\infty} f_m(t)e^{-st}dt$. Let *t* be the time interval between two consecutive calls from the MBS to the user, and t_1 be the time interval between the first call and the time when the user moves to a new RA. From the random observer property of the arrival call stream [4], if call arrivals are Poisson distributed and *F(t)* is an exponential distribution, the hit ratio of a user calling can be given by $p = \int_{t=0}^{\infty} \lambda e^{-\lambda t} \int_{t_1=t}^{\infty} f(t_1)dt_1 dt$, where in exponentially distributed with parameter μ. That is, $f(t_1) = \mu e^{-\mu t_1}$, and $F(x) = 1 - e^{-\mu x}, x \ge 0$. From these relationships, we can express the hit ratio as follows [7].

$p = \int_{t=0}^{\infty} \lambda e^{-\lambda t} \int_{t_1=t}^{\infty} \mu e^{-\mu t_1} dt_1 dt = \lambda/(\lambda+\mu)$…………….. (3)

Note that for different values of LCMR, there will be different values of hit ratio.

3. The FBFind Algorithm

Next, we propose a modified algorithm that is based on the Per-User Location Caching algorithm of Jain et al [6]. In our algorithm, we specify all the previously introduced parameters and add some more to account for further details. The basic idea behind using FBFind algorithm is to reduce the SS7 message traffic and database access traffic by maintaining a local storage[10]. The local storage is called Location Information Library (LIL), and each LSTP maintains a LIL. A LIL is a table lookup that maintains entries of recently visited locations information. The FBFind method uses LIL entries that are stored in the LSTP to determine the location of a user. Each LIL has at least *n* number of entry lists, which are frequency based information of MHs in the LSTP. When a call is made to an MH, the algorithm checks whether the called is in the same RA or not. If the called is in the same RA, there is no need to perform any update since the respective MBS already has knowledge of both the callee and the caller. When the callee and the caller are not in the same RA, the algorithm is called into action to take the following steps.

1. When an MH is called and it is not in the same RA, and there is an entry for this MH in the LIL, then the FBFind Algorithm must be employed. Go to Step 3 in this case.

2. If there is no LIL entry for that MH, then go to Step 6 where the Basic Find algorithm is executed.
3. Select the LIL entry, V_L, with the highest request frequency in the list. The current MSB of the caller queries the location of the called V_L.
4. If the caller MSB gets the routing information of the called from V_L, FBFind results in a hit, and the connection between the caller and called is established.
5. If the caller MSB does not get the routing information from V_L, FBFind results in a miss.
6. Otherwise, the Basic Find Algorithm[6] is employed to find the called MH.

As was stated before, each LSTP has a storage, referred to as Location Information Library (LIL). A LIL stores the mobility patterns information of the MHs which visit that LSTP. This information is retrieved during a call delivery. In a distributed information library architecture with multiple LILs, it is beneficial to avoid unnecessary performance bottlenecks because we do not need to go to the HLR of the caller and get the information about it. As illustrated in Figure 4, the home location register (HLR) of MH_A is MBS_A and the HLR of MH_B is MBS_B and so on. Likewise, LIL_A has the location and routing information of MH_A, LIL_B has those information of MH_B, LIL_C has those information of MH_C, and so on. Clearly, LIL_A and LIL_B should not be in $LSTP_1$ because MBS_A and MBS_B are under $LSTP_1$, and they have the information on both MH_A and MH_B respectively.

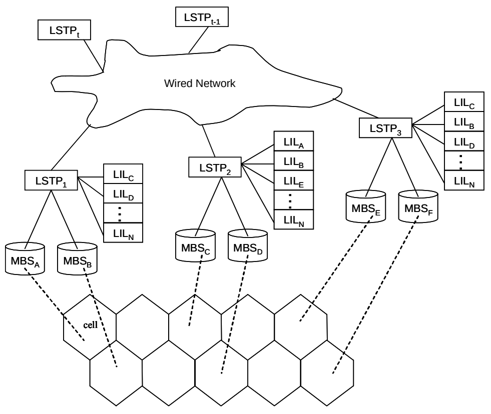
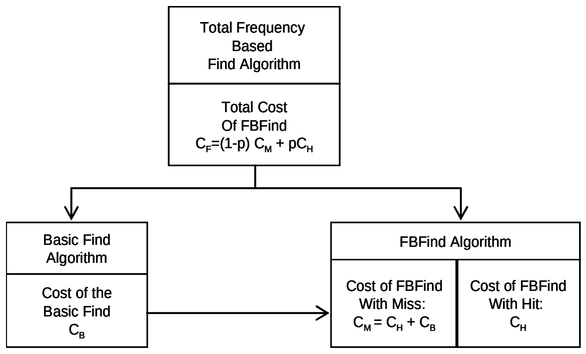


Figure 4. System Model

4. Evaluation of the FBFind Algorithm

We evaluate the FBFind algorithm introduced in the previous section in terms of four parameters that will identify various cost functions associated with the algorithm implementation. We can relate the four parameters to the algorithms two parts: Basic Find algorithm which is associated with the cost of the Basic Find (C_B), and FBFind algorithm which is associated with three costs: FBFind with miss (C_M), FBFind with hit (C_H) and FBFind total cost (C_F). Figure 5 depicts the steps and relationships between these four parameters.



Total Frequency Based Find Algorithm
Total Cost Of FBFind C_F=(1-p) C_M + pC_H
Basic Find Algorithm
Cost of the Basic Find C_B
FBFind Algorithm
Cost of FBFind With Miss: $C_M = C_H + C_B$
Cost of FBFind With Hit: C_H

Figure 5. The Four Cost Parameters Associated with FBFind and The Relationships

4.1. Cost of the Basic Find Algorithm, C_B

We recall that the Basic Find algorithm is based on the IS-41 Basic HLR/VLR Scheme [2, 11]. Also Basic Find algorithm is introduced by Jain et al [6]. In here, we extended and detailed their algorithm one step further. As such, the HLR maintains location information for the user. When a user travels to a new RA, his HLR is updated and an entry in the VLR is made. When a call is placed to locate this user, the caller's MBS first checks the local VLR for the callee's profile.

The steps and the pseudocode to find the cost of Basic Find (C_B) are as follows.

1. The call to the MH_j is directed to MBS_2. Assuming that the callee is not located within the same RA, the MBS_2 queries the HLR of MH_j, MBS_7, for routing information.
2. The HLR of MH_j, MBS_7, contains a pointer to the VLR, MBS_{15} in whose associated RA the caller party is currently situated.
3. The VLR in turn queries the MBS of MH_i to check if the callee is available.
4. The MBS_7 returns a routable address to the VLR of MH_j, MBS_2, if callee is available.
5. The VLR should send routable address to the HLR of MH_i.
6. The HLR of MH_i forwards the address to the MBS of the caller.

Steps and the system model used is shown in Figure 6. In this figure, we use the SS7 model [10] to show how to find the cost of the Basic Find algorithm [6]. The signaling network cost incurred in locating a user in the event of a call is the sum of the cost of querying the HLR and the cost of querying the VLR. Let Q_H be the cost of a query to the HLR to obtain the current VLR location; Q_V be the cost of a query to the VLR to obtain the routing address; Q_A be the cost of a query to the HLR and receiving a response, and Q_B be the cost of querying the pointed to VLR and receiving a response. The calculation of the Basic Find costs of the worst case is as follows.

$C_B = Q_A + Q_B = 2(R_A + R_B + R_C + P_P + R_C + P_R + R_B + P_L + R_A) + Q_H + 2(R_A + R_B + R_C + P_P + R_C + P_R + R_B + P_L + R_A) + Q_V$

Under the assumptions that $R_A = R_B = R$, and $P_R = P_L = P$, the expression above can be written as, $C_B = 16R + 8R_C + 4P_P + 8P + Q_H + Q_V$ ……………………………………(4)

4.2. Cost of the Total FBFind Algorithm, C_F

The cost of the total Frequency-Based Find algorithm (C_F) can be defined as the sum of the cost of the FBFind with hit (C_H) times hit ratio (p) and the cost of the FBFind

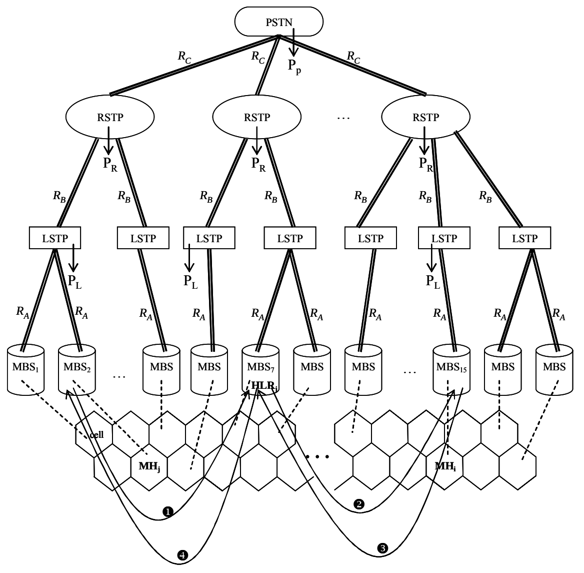


Figure 6. Basic Find Algorithm Steps: Callee MH$_j$ is calling a caller MH$_i$

with miss (C_M) times the miss ratio (1 – p). We define the hit ratio as the relative frequency of getting the correct user's location when it is consulted in the LIL entries. In Section 2.3, we defined the hit ratio as $p = \lambda/(\lambda+\mu)$, while $LCMR = \lambda/\mu$. Then we have,

$C_F = pC_H + (1\text{-}p)C_M$ …………………………………...(5)

Let us define the Cost of the FBFind with Hit. Let Q_L and Q_R be the cost of querying (query and response) a local LSTP and a remote LSTP respectively; and *q* be the probability of getting the MH's information on a local LSTP's LIL. Also, as before we define Q_H as the cost of a query to the HLR to obtain the current VLR location, and Q_V as the cost of a query to the VLR to obtain the routing address. Then, $C_H = q(Q_L + Q_V) + (1 - q)(Q_R + Q_V)$, where, $Q_L = 2R_A = 2R$ since it is necessary to go the LSTP and come back, and $Q_R = 2(R_B + R_C + P_P + R_C + P_R + R_B + P_L + R_A) = 6R + 4R_C + 2P_P + 4P$. Substituting these values, we obtain:

$C_H = 6R + 4R_C + 2P_P + 4P + Q_V - q(4R + 4R_C + 2P_P + 4P)$..(6)

The cost of the FBFind with miss (C_M) algorithm can be determined by adding the cost of the FBFind with hit plus the cost of the Basic Find algorithm because miss occurs there is no entry for that specific MH after all LIL entries are checked.

$C_M = C_H + C_B$ …………………………………………(7)

Finally, we substitute C_H, and C_M into equation (5), then:

$C_F = pC_H + C_M - pC_M = C_H + C_B - pC_B$ ………………(8)

$C_F = 22R + 12R_C + 6P_P + 12P + 2Q_V + Q_H - q(4R + 4R_C + 2P_P + 4P) - p[16R + 8R_C + 4P_P + 8P + Q_H + Q_V]$ ………………(9)

4.3. Evaluation of the Total FBFind Algorithm

Let us simplify little more for each of cost equation. We have two equations (1) and (2), $R_C = dR$ and $P_P = dP$ respectively. We can assume that t, the total number of RSTP, be any numbers. Let t be 6 so that we can have integer value of d, this case d equals to 4. Then we have $R_C = 4R$ and $P_P = 4P$. Now we substitute these results into equation (4), the cost of the Basic Find Algorithm, and equation (9), the cost of the total FBFind algorithm then we have following two new equations.

$C_B = 48R + 24P + Q_H + Q_V$ …………………………(10)

$C_F = R(70 - 20q - 48p) + P(36 - 12q - 24p) + Q_V(2 - p) + Q_H(1 - p)$ ………………………………………(11)

Note that in equation (11), Q_V and Q_H depend only on the value of LCMR. If we have lower LCMR value, that will correspond to higher Q_V, and Q_H.

Now, let's take a look at the hit ratio, *p*. We have the equation for hit ratio in Section 2.3, and they are: $p = \lambda/(\lambda+\mu)$, and $LCMR = \lambda/\mu$. So $\lambda = \mu LCMR$ then,

$$p = \frac{\mu LCMR}{\mu LCMR + \mu} = \frac{LCMR}{LCMR + 1} \quad \text{………………………...(12)}$$

From this equation, we can have the relationship between LCMR and other parameters. We substitute equation (12) into equation (11), and rearrange the result equation as the simple form:

$C_F = aR + bP + c$, where $a = 70 - 20q - 48p$, $b = 36 - 12q - 24p$, $c = Q_V(2 - p) + Q_H(1 - p)$ ……………………(13)

We can have different values of local call-to-mobility ratio, LCMR, and *q*, the probability of getting the MH's information on the local LSTP's LIL. Table 2 shows the results of different values of LCMR and q. With these results, we can say few things about our algorithm. When we have the same LCMR, we have lower cost with higher q. When we have the same q, we have lower cost of FBFind with higher LCMR.

Table 2. Coefficients of C_F when d = 4

LCMR	*p*	*q*	*a*	*b*	*c*
1	0.5	0.1	44.0	22.8	1.5 Q_V + 0.5 Q_H
5	0.8	0.1	28.0	14.8	1.2 Q_V + 0.2 Q_H
10	0.9	0.1	24.4	13.0	1.1 Q_V + 0.1 Q_H
1	0.5	0.5	36.0	18.0	1.5 Q_V + 0.5 Q_H
5	0.8	0.5	20.0	10.0	1.2 Q_V + 0.2 Q_H
10	0.9	0.5	16.4	8.2	1.1 Q_V + 0.1 Q_H
1	0.5	0.9	28.0	13.2	1.5 Q_V + 0.5 Q_H
5	0.8	0.9	12.0	5.2	1.2 Q_V + 0.2 Q_H
10	0.9	0.9	8.4	3.4	1.1 Q_V + 0.1 Q_H

5. FBFind and Caching Algorithms: A Comparative Study

In this section, we compare two algorithms: our FBFind Algorithm and Jain's Caching Algorithm [6]. The caching algorithm has scheme for very similar defining the cost of Basic Find and the total cost. The main difference between these two algorithms is the cost for hitting. We use the same assumptions for the two algorithms under comparison as those used in FBFind algorithm in previous sections. The assumptions are: $R_A = R_B = R$, $P_R = P_L = P$, $R_C = dR$, $P = dP$, $Q_L = 4R + 2P$, and $Q_R = 4R + 2dR + dP + 2P$, where, d = 4cos(180/t)/sin(360/t) = 4. Let's take a look at these parameters for the caching algorithm next.

5.1. Calculations of Parameters in the Caching Algorithm

The cost of the Basic Find in caching algorithm is the same as FBFind one equation (10), and that is,

$C_B = 48R + 24P + Q_H + Q_V$ …………………………(14)

The cost of the caching algorithm with hit is different from that for FBFind with hit since there is no local storage for the caching algorithm. In this case, the cost function can be written as $C_{Hc} = q(Q_{Lc} + Q_V) + (1 - q)(Q_{Rc} + Q_V)$, where, $Q_{Lc} = 4R_A + 2P_L$ and $Q_{Rc} = 2(R_B + R_C + P_P + R_C + P_R + R_B + P_L + R_A) = 6R + 4R_C + 2P_P + 4P$ Substituting these values in the C_{Hc} equation above, we obtain.

$C_{Hc} = 22R + 12P + Q_V - q(18R + 10P)$ …………..(15)

Then the total cost of caching algorithm C_C is, $C_C = C_{Hc} + C_B - pC_B$. Substituting the values of C_{Hc} and C_B in equation (14) and (15), we have

$C_C = R(70 - 18q - 48p) + P(36 - 10q - 24p) + Q_V (2 - p) + Q_H (1 - p)$……………………………..…………………(16)

This equation can be rewritten using the same form as equation (13).

$C_C = eR + fP + g$ ……………………………………(17)
with, $e = 70 - 18q - 48p$, $f = 36 - 10q - 24p$, and $g = Q_V (2 - p) + Q_H(1 - p)$

As we can see in equation (17), the value of g for both FBFind and cache algorithms are the same, i.e., c = g. The coefficients of the C_C are summarized in Table 3 for different values of LCMR, p, and q.

5.2. Algorithms Comparisons

Equations (11) and (16) were derived for a fixed value of *d* since this *d* affects both algorithms similarly. This means that if we change the value of *d* for both algorithms and compare them, we will have the similar result for both. However, if we compare FBFind algorithm with other algorithms other than caching, we may not want to fix this parameter. By examining Tables 2 and 3, we notice that coefficients of the equations, i.e., a, b, c, e, f, and g, are closely related to the parameter *q*, the probability of getting the MH's information on local LSTP's LIL. If we have higher *q* values, we have more cost saving for FBFind algorithm under the same LCMR. Figure 7 shows the comparison between FBFind algorithm and caching algorithm. When *q* equals to 0.1, the gap between these two algorithms is 0.2. But when *q* is 0.9, the gap increases to 1.8. This tells us that the FBFind algorithm works better with higher *q*. Now let's look at the efficiency for different values of LCMRs. We define the efficiency of cost saving (*E*) over the caching algorithm as follows:

$$E(\%) = \frac{(\text{Caching\_Cost}) - (\text{FBFind\_Cost})}{(\text{Caching\_Cost})} \times 100$$

Figure 8 shows the efficiency with three different values of LCMRs. When LCMR is 1, FBFind has about 0.5% better saving at *q* is 0.1. But when we have *q* of 0.9 the savings jump to 6% better efficiency than caching. In the case of LCMR is 10, FBFind has 0.8% better efficiency at *q* = 0.1 but at *q* = 0.9, we have about 18% of better efficiency. The overall behavior indicates that the FBFind algorithm works best with higher LCMR and higher *q*.

Table 3. Coefficients of C_C when d = 4

LCMR	*p*	*q*	*e*	*f*	*g*
1	0.5	0.1	44.2	23.0	$1.5\ Q_V + 0.5\ Q_H$
5	0.8	0.1	28.2	15.0	$1.2\ Q_V + 0.2\ Q_H$
10	0.9	0.1	24.6	13.2	$1.1\ Q_V + 0.1\ Q_H$
1	0.5	0.5	37.0	19.0	$1.5\ Q_V + 0.5\ Q_H$
5	0.8	0.5	21.0	11.0	$1.2\ Q_V + 0.2\ Q_H$
10	0.9	0.5	17.4	9.2	$1.1\ Q_V + 0.1\ Q_H$
1	0.5	0.9	29.8	15.0	$1.5\ Q_V + 0.5\ Q_H$
5	0.8	0.9	13.8	7.0	$1.2\ Q_V + 0.2\ Q_H$
10	0.9	0.9	10.2	5.2	$1.1\ Q_V + 0.1\ Q_H$

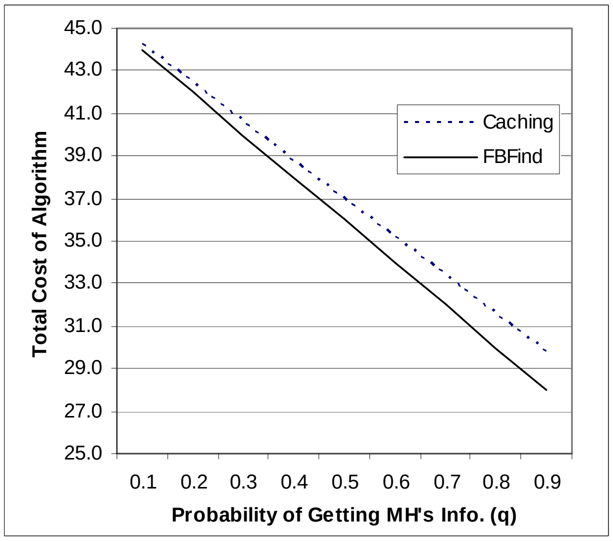


Figure 7. Cost Functions with Respect to Values of q

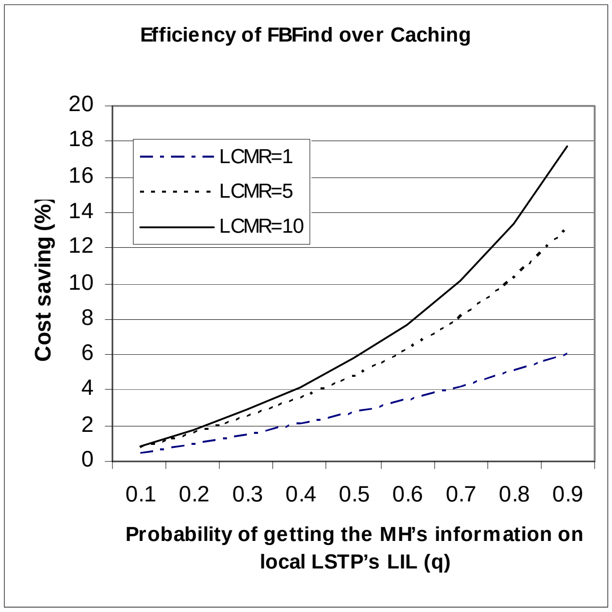


Figure 8. Efficiency Comparisons

6. Conclusions

To further the research in the area of location and mobility management, the Frequency-Based Find (FBFind) algorithm is introduced in this paper. This algorithm saves some costs by using local storage called Location Information Library (LIL). The scheme requires each LSTP to maintain a LIL. A LIL is a table lookup that maintains entries of recently visited location information, thereby saving on unnecessary routing and querying costs. In addition, the paper introduces a new modeling technique which works with the FBFind algorithm. This technique uses hierarchical binary tree, up to the RSTP level, to model the system architecture. Based on the simulation results obtained, we show that the performance of FBFind algorithm depends heavily on two conditions: local call to mobility ratio and the probability of getting the mobile host's information on local LSTP's information library. We observed that the FBFind algorithm can save on the cost of searching for information by 6% to 18% over different values of LCMR between 1 to 10.

We also compared the FBFind algorithm with the well known Caching algorithm. However, if we compare FBFind algorithm with other algorithms than caching, we may not want to fix this parameter since we cannot guarantee that this parameter will affect both algorithms in similar way. As can be seen from our results, if we have low LCMR and a MH travels many cells within short period of time, the FBFind algorithm will not give good results. As an extension of the existing work, architectures with different numbers of RSTPs, LSTPs and MBSs, and topologies as well as models will be studied.

References

[1] Bharghavan, V., "Challenges and Solutions to Adaptive Computing and Seamless Mobility over Heterogeneous Wireless Networks," Wireless Personal Communications, Vol. 4, pp. 217 – pp. 236, 1997.

[2] EIA/TIA, Cellular radio-telecommunications Intersystem Operations, Technical Report IS-41 Revision B, EIA/TIA, 1991

[3] Elnahas, A., and N. Adly, Location Management Techniques for Mobile Systems, Information Sciences, Vol. 130, pp. 1 – 22, 2000.

[4] Feller, W., An Introduction to Probability Theory and Its Applications, Wiley, New York, 1996.

[5] Goodman, T.J., Trends in cellular and cordless communications, IEEE Communications Magazine Vol. 29, No. 6, pp. 31 – 40, 1991.

[6] Jain, R. Y. B. Lin, C. Lo, and S. Mohan, A Caching Strategy to Reduce Network Impacts of PCS, IEEE Journal on Selected Areas in Communications, Vol. 12, No. 8, pp. 1434 – 1444, 1994.

[7] Jain, R., and Y.B. Lin, An Auxiliary User Location Strategy Employing Forwarding Pointers to Reduce Network Impacts of PCS, Proceedings International Conference on Communications, pp. 1 – 26, 1995.

[8] Krisha, P, N. H. Vaidya, and D. K. Pradhan, Static and Adaptive location Management in mobile Wireless Networks, Computer Communications, Vol. 19, pp. 321 – 334, 1996.

[9] La Porta, T. F., M. Veeraraghavan, P. A. Treventi, and R. Ramjee, Distributed Call Processing for Personal Communications Services, IEEE Communications Magazine, Vol. 33, No. 6, pp. 66 – 75, 1995.

[10] Modarressi, A. R., and R. A. Skoog, Signaling System No. 7: A Tutorial, IEEE Communications Magazine, pp. 19 – 35, July 1990.

[11] Mohan, S., and R. Jain, Two User Location Strategies for Personal Communications Services, IEEE Personal Communications, Vol. 1, pp. 42 – 50, First Quarter 1994.

[12] Pierre, S., Mobile Computing and Ubiquitous Networking: Concepts, Technologies and Challenges, Telematics and Informatics, Vol. 18, pp. 109 – 131, 2001.

[13] Suh, B. S., J. S. Choi, J. K. Kim, Design and Performance Analysis of Hierarchical Location Management strategies for Wireless Mobile Communication Systems, Computer Communications, Vol. 23, pp. 550 – 560, 2000.

[14] Tabbane, S., Location Management Methods for Third-Generation Mobile Systems, IEEE Communications Magazine, Vol. 35, No. 8, pp. 72 – 78, 1997.

[15] Weng, C. M., and P. W. Huang, Modified Group Method for Mobility Management, Computer Communications, Vol. 23, pp. 115 – 122, 2000.