Mobile Information Systems 5 (2009) 313–337
DOI 10.3233/MIS-2009-0086
IOS Press

Mobility management algorithms for the Client-driven Mobility Frame System – mobility from a brand new point of view

Péter Fülöp, Benedek Kovács and Sándor Imre
Department of Telecommunications, Budapest University of Technology, H-1521 Pf. 91, Hungary
E-mail: {fpeti,bence,imre}@mcl.hu

Abstract. In this paper a new mobility management is introduced. The main idea in this approach is that the mobil node should manage the mobility for itself not the network. The network nodes provide only basic services for mobile entities: connectivity and administration. We construct a framework called the Client-based Mobility Frame System (CMFS) for this mobility environment. We developed the CMFS protocol as a solution over IPv4 and we show how to use Mobile IPv6 to realize our concept. We propose some basic mobility management solutions that can be implemented into the mobile clients and give details about a working simulation of a complete Mobility Management System. Example mobility management approaches such as the centralized- and hierarchical- or cellular-like ones are also defined and hints are given what kind of algorithms might be implemented upon the Client-based Mobility Frame System over IPv4 and IPv6 as well. We introduce some example algorithms that can work with the CMFS making mobility management efficient by minimizing signalling load on the network. In the present work modeling and detailed discussion on the parameters of the algorithms is given and comparison to existing mobility approaches and protocols is done. We prepared a simulation to test our protocol and to back up the proposals we provide the reader with simulation results. We stress that still one the most important benefit of our findings is that all the MNs can run different management strategies and can optimize mobility for themselves.

1. Introduction

Seamless information mobility is a requirement in today's world. Although there are many other alternative operating solutions there is still a need for IP mobility since IP is the most widespread protocol. The communicating equipments are identified with their permanent IP address and the communication is done on IP networks. Many works have discussed the problem of managing the movement of the clients, since the Internet was designed to be static and does not support mobility by itself. There are different solution proposals for the problem and all of them have their drawbacks and good features. If one takes a close look at these systems they always deal with the tradeoff between complexity (simplicity) and optimality. Naturally, this can not be resolved, but we will transform it into another dimension: from network level to mobile node level.

In this paper we present our agent based mobility management solution and propose a couple of strategies one can implement to the mobile equipment and then have very efficient mobility solution. Our alternative point of view is easy to implement and does not require complex network setups to operate. We do not say that we have found the optimal system to provide IP or other kind of mobility, but we came up with a brand new idea and framework which is very different from the classical approaches and can be the most cost-efficient in many cases.

The basic idea is that, unlike in the GSM or classical Mobile IP solutions, the network will no longer have to provide any logic for the management algorithm. The whole can remain simple and the nodes will only have to handle simple commands or simple protocols (like Mobile IPv6 – MIPv6 [8]) by recognizing, executing and forwarding simple messages generated by the mobil entity itself. The management system is implemented in the mobile client, accordingly each node is able to choose the most suitable mobility for itself on the same network. We show how to apply the classical strategies like the simple MIP, hierarchical, tracking or cellular approaches to our system and furthermore we propose an algorithm that creates cell structures efficiently and individually for each MN.

The structure of the paper is the following. First we will present some related work and our idea, then give example protocols, i.e. we define the Client-based Mobility Frame System, and than we give mobility applications what is the Mobility Management Systems itself. Later we present smart mobility management strategies and compare them to classical approaches. The implementation issues over IPv4 and IPv6 are discussed in Section VI and finally simulation and numerical results are given in Section VII and in Section VIII.

2. Related work

As previously mentioned many solutions have been developed to fill the gap for ip protocol mobility support [10]. The most widely know, first solution was the Mobile IPv4 (MIP) [9], which was developed for IPv6 (MIPv6) [8,25] as well. The basic idea is if the mobile node leaves its home network and connects to a foreign network then it will be assigned a care of address. The home agent, which represents the mobile node in the home network, when the node is away, will be informed. In such cases the home agent tunnels all messages to the mobile node's care of address. This leads to the creation of a costly triangle communication, which inspired the development of different route optimization solutions, including MIPv6 [8,25] at first.

Later on newer, improved versions of the mobile ip appeared, including the Hierarchical Mobile IP (HMIP) [6,7] and the Dynamical Hierarchical Mobile IP (DHMIP) [26]. The basic idea of these protocols is that instead of the global management system a local system is used to minimize the network signaling traffic. Certainly other solutions were published besides Mobile IP, like Cellular IP (CIP) [2] which supports fast, local mobility or Host Identity Protocol (HIP) [20], which separates the IP address's global identifier and localizing function.The Location Tracking (LTRACK) [5] developed for low call ratio and the hierarchic version of the paging solution, Hierarchical Paging (HPAGE) [13], known from CIP are also improvements of the earlier published mobility protocols [3,10]. Application level solutions like Session Initiation Protocol (SIP) [11] were also born.

Papers describing the integration of the existing protocols were also published, like the SIP and MIP two layered mobility [12], which provides fast hand-off and results in decreased global signaling traffic. The previously introduced Home Agent Architecture's adequate implementation [14] decreases the time of location update and the time of tunneling.

Usage of any of the above in a network does not necessarily result the most optimal solution for each mobile node. Our proposal is that the mobile node could select the most optimal mobility support for itself based on its own mobility parameters, movement intensity or the visited network sub-tree.

3. Client-driven mobility frame system (CMFS)

In this Section we will introduce our concept, the Client-driven Mobility Frame System (CMFS from now) in detail. We specify the basic roles of the network and what capabilities the fix network nodes are

required to have to be able to communicate with the moving entity. A simple method will be given for the mobile node to discover the service network and build up its own logical network.

It is essential that the network nodes understand the commands from the mobile client that will be the brain and the director of a given mobility management system itself. Even each client can use different management strategies on the same network with the same technology and capabilities.

3.1. The idea

The client driven mobility management we introduce is inspired and based (but not depend!) upon the fact that a mobil user typically moves within a range of access points and rarely leaves to far away agents. In order that the mobile could manage its own mobility it has to maintain a database of the nodes it communicates with. This is called the Logical Network (LN). The mobile node should always be able to have an up-to-date information of the nodes of this network. The size of this depends on the algorithm the mobil uses.

To give an example we will show later that to implement a basic Mobile IP-like solution on our framework system the mobile only has to maintain information about 3 (or even only 2) nodes in the network, so for a node with very limited capacity this can be a solution good enough with a cost similar to the cost of Mobil IP. On the other hand, more complex entities like PDAs can maintain information about thousands of MAs and thus optimize communication costs or quality.

The most important advantage of our solution is that the service providers do not have to choose an exact mobility approach, which could be very inefficient in certain cases for certain users. The mobile nodes in CMFS can choose the optimal algorithm for themselves, thus the mobility solution can be the most cost-efficient and adaptable for various circumstances.

One more thing we have to point out is that all the MNs in the world will see different networks and optimize the mobility for themselves. For this reason we believe that our solution has a better network resource utilization than any other classical one.

3.2. Basic notations and requirements

Since there are many kind of notations in this field, to avoid misunderstandings, some of the basic ones we use are defined here. The mobility model will be the same abstract one as in one of our previous works [16,18]:

- The Mobile Nodes (MN, alias mobiles, moving entities) are the mobile equipments who want with communicate to any other mobile or fixed partner.
- There are Mobility Access Points (MAP) as the only entities who are capable to communicate with the Mobile Equipments, have the CMFS implemented (Note: mobility does not necessary imply radio communication. It means only that the Mobile Node changes its Mobility Access Points and when it is attached to one, communication between them can be established).
- The Mobility Agents (MA) are network entities running the mobility management application.
- There is a core network that provides communication between the Mobility Access Points and has a structure that can be described with a graph. Vertices are either Mobility Access Points or Mobility Agents other serving nodes who are not part of the mobility management application and the edges can be any kind of links (even radio links) for the data communication between the vertices.
- Home Agent (HA) a special MA that is a kind of basis to the Mobile just like in the Mobile IPv4 [9] or the Mobile IPv6 [8] case or the HLR in the GSM case. Whenever the MN is paged its exact or approximate location can always be found in the database of this node.

To be able to serve the MNs and their algorithms we define requirements for the network. Once the MAs are aware of all these the MN can use any kind of Mobility Management Strategy (MMS) for itself. This also means that different terminals are allowed to use the most suitable MMS for themselves.

The most important is that all the nodes in the network should be able to communicate with each other i.e. they find each other with some unique identifier (e.g. IP address). This is a very basic requirement and it is provided in even analogous PSTNs, IP or most communication networks.

Another important statement is that CMFS does not care about the access technology of the MN. This should be provided by the network. It does not matter whether the MN connected via WLAN or GSM or even MIPv4 or MIPv6 specified access. The only thing is that the MN should be registered to a MAP in the CMFS what is considered as its connection point. Handover decision algorithms between the MAPs in our CMFS will be implemented in the MN and are discussed later.

The good thing in the requirements above is that the CMFS protocol is totally independent of all the underlaying technologies no matter whether the communication goes over IP (BGP, OSPF, IS-IS), ATM (IISP, PNNI) in a GSM (TCAP) environment or anything else; the important thing is that the nodes find each other.

Note that not all the nodes in the network must necessarily have the CMFS implemented on, but only the ones will be recognized by MN.

The HA should always know where to route a packet towards the MN, or drop the call. There should be a databaseregistry for this for example an association between the MNs permanent IP (Care of Address, CoA) and routes: where to forward the packet towards the CoA. All the MAs should work in a similar way. Once the MN with its CoA paged at an MA it should route the packets to the MAP of the MN. If there is no route to the MN it should simply drop the packets.

3.3. The CMFS in the mobile node

We have seen what properties a network should have to be able to adopt our solution. Now we define the kind of logic that should be implemented into the network. We do it in the following way. At first we show how the MN can find out the network structure and then we give an explicit specification of the commands i.e. the messages the MN sends to the network nodes and the actions they take.

3.3.1. Network discovery – logical network

The first task for the MN is to discover the network. It can send a message to any of the MAs it knows (the MN has to know at least one MA, the one that plays a Mobile IP Home Agent (HA)-like role). Also in most cases the MN is in a Foreign Network and thus attached to a MAP. Sending the message to the HA through the Network all the nodes the message passes by should reply with their IDs (IP address if it is an IP network) to the MN, so that it could calculate their logical position by the delay data. This is how the MN builds a Logical Network (LN) it uses as an input for its algorithm.

The solution we have chosen in our implementation is an IP based one and uses the IP traceroute packets. By using small TTL values which quickly expire, *traceroute* causes that the routers along a packet's normal delivery path automatically generate an ICMP Time Exceeded message. The costs of using these links are determined by measuring the delay.

The Logical Network structure for the MN must not be the same as the real network topology. Since the Logical Network is built up with measurements we think that it gives a better view of the real status of the network. For example if a node is down, it does not reply to the MN, i.e. the MN thinks that it does not exist, what is actually true from the MN's point of view since the MN can communicate with operating nodes only.

3.3.2. Messages and actions

The main element of our solution is that the MN orders the MAs to modify a kind of routing database they maintain. The database in MA has entries like: "The MN can be reached via the MA_i or MAP_i"; so if a packet for the MN arrives to a given MA, it checks if there is an entry in its database and routes according to the rule it finds. If there is no such route, the packet has a correct destination on the mobility level, only routing provided by the underlaying network is needed. The MA and MAP should be capable to register and delete new entries from the database upon the commands from the MN. Also they have to be able to forward such messages to each other ordered by MN.

How can a MN instruct an MA? When it attaches to a new MAP after a successful handover it naturally registers there. Let us construct such a register message structure:

[Dst: MAP_i, Src: MN, Actions: Register MN to MAP_i via MN;
[Dst: MA_j, Src: MAP_i, Actions: Register MN to MA_j via MAP_i,MAP_{ii},MAP_{iii};
[Dst,Src,Actions: ; ;
[. . .
[Dst: HA, Src: MA_n, Actions: Register MN to HA via MA_n]
]
]
]].

As you can see, these registration instruction are embedded into each other. As we mentioned above the action in the structure can be a *Register*, a *Delete*, or another instruction structure, which has to be forwarded by the actual MA. What the MA should do is to understand this message and maintain the entry in its database: if the paged node is MN then it should be searched via HA, MA_n, . . . , when MN is searched at MA_j then MA_j knows that it can be reached via MAP_i, MAP_{ii}, MAP_{iii} meaning that the packet is routed to all the 3 nodes representing a CIP-like algorithm [2]. If there is no such multiple route and the messages does not always contain the HA, then HMIP-like [6] is implemented. If there is an update that goes from MN directly to HA, then MIP-like [9] is implemented, if these last two kind of messages are mixed then a DHMIP-like [26] approach is presented. If the node sends messages like

[Dst: MAP_i, Src: MN, Actions: Register MN to MAP_i via MN;
[Dst: MAP_{i-1} //The former node//,
Src: MN, Actions: Register MN to MAP_{i-1} via MAP_i
]]

then a HAWAII-like protocol [21] is implemented. In case of wireless tracing (for example LTRACK [5]), two different messages would be sent:

[Dst: MAP_i, Src: MN, Actions: Register MN to MAP_i via MN;
]
[Dst: MAP_{i-1} //The former node//,
Src: MN, Actions: Register MN to MAP_{i-1} via MAP_i]

We have provided an example implementation that can be modified after reasonable discussions but just like IP or any kind of protocol it should be standard in any network the MN wants to communicate in.

4. The MN based management strategies

We have shortly introduced the new idea and explained the basics of its operation. In this section we will show possible strategies the MN can use to make the mobility management work. We will present different approaches here and discuss them separately, but note that a single MN can use multiple of these depending on its location for example.

Suppose that mostly we are working in the university and we spend most of the day in our room and in two labs thus stay under a set of agents and access points. However, in the afternoon and in the morning we travel long distance passing through many MAPs. At home we have a router that is our HA. Intuitively we can think that the cellular approach is useful at the university and a tracking-like solution is the most efficient on the way home.

Example mobility management strategies in CMFS are coming now.

4.1. Mobility management strategies using CMFS, inspired by classical solutions

Here we show how to implement the versions of the basic ip mobility protocols, such as hierarchical, cellular, Mobile IP etc. on the CMFS system.

4.1.1. Personal mobile IP – PMIP

The operation of Personal Mobil IP is simple and easy and has been already discussed in our former papers [19]. Once the MN attaches to MAP it registers himself to the HA. The operation is very similar to MIP and has a great advantage. The MN has to make no extra computation and has to maintain no extra database while there are always a few routes in the MAP.

[Dst: MAP_i, Src: MN, Actions: Register MN to MAP_i via MN;
[Dst: HA, Src: MAP_i, Actions: Register MN to HA via MAP_i
];
[Dst: MAP_{i-1}, Src: MAP_i, Actions: Delete MN in MAP_{i-1} via MN
]].

Where the second message is only needed if clearing the network is up to the MN unlike in MIPv4. This solution is referred as pure PMIP (P-PMIP).

The simple PMIP protocol operates like MIP and has approximately the same capacity consumption as well as we will later see. We would like to point out that the MN has to maintain a Logical Network of 3 nodes only. However, a great benefit of our proposal is that any MN can implement different version (e.g. soft handover) of the protocol without any modification in the network entities.

Then the Extended PMIP (E-PMIP) is an example of extension of PMIP when there is no packet loss and no obsolete routes in the databases of the MAs but of course the messages are more complex. One can see what happens in case of a handover on Fig. 1.

[Dst: MAP_i, Src: MN, Actions: Register MN to MAP_i via MN;
[Dst: MAP_{i-1}, Src: MAP_i, Actions: Register MN in MAP_{i-1} via MAP_i;
Delete MN in MAP_{i-1} via MN;
[Dst: HA, Src: MAP_{i-1}, Actions: Register MN to HA via MAP_i;
Delete MN in HA via MAP_{i-1};
[Dst: MAP_{i-1}, Src: HA, Actions: Delete MN in MAP_{i-1} via MAP_i
]
]
]].

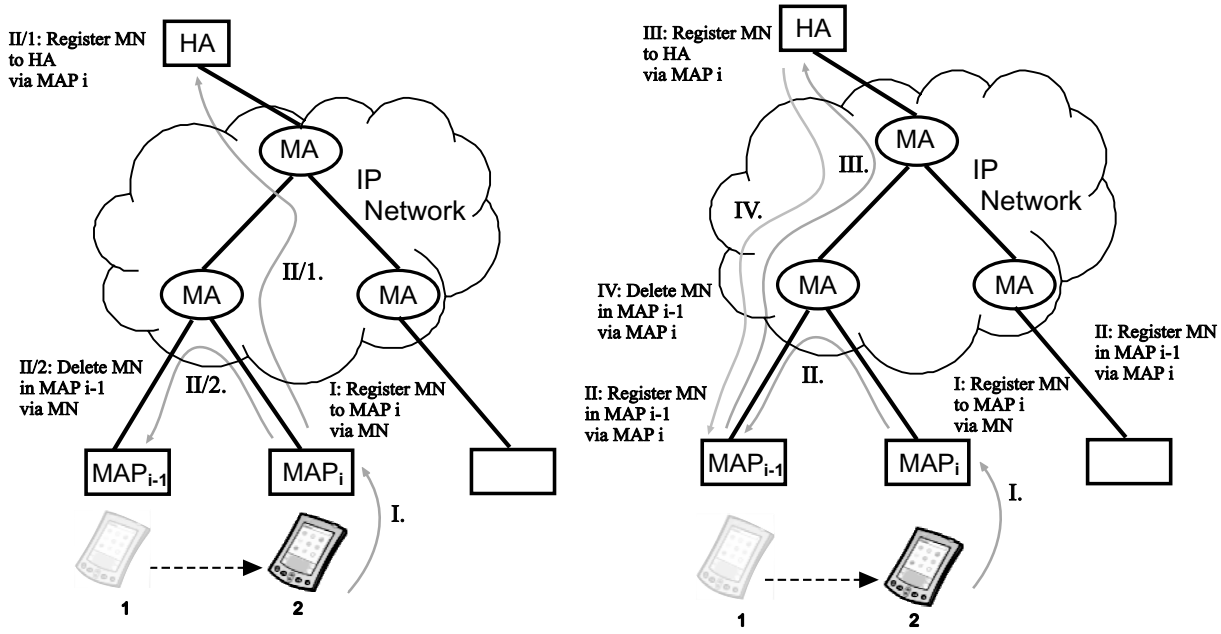

Fig. 1. On the left hand side figure one can see the basic P-PMIP protocol, while the right hand side depicts the operation of the action-linearized Personal Mobile IP Mobility Management System (E-PMIP) with soft handover mechanism.

4.1.2. Personal hierarchical mobile IP – PHMIP

In the classical hierarchical Mobile IP there is a fixed network architecture that is a tree of nodes of Mobility Agents as Hierarchy Points (HP) and leaves of Mobility Access Points. The MN updates only to the nearest HP thus saving signalling load on the HP-HA route.

The operation of a HMIP micro-mobility (talking about only two layered hierarchy) would give us the question: which node should be the MA in the hierarchical mobility approach. We suppose that seeing the traceroute messages, the MN can decide it. After that the command messages are again simple and easy to construct.

More problem arise when talking about multiple layered hierarchical solutions. The MN has to make complex calculations setting up the network tree, but still the only problem will be to locate the logical junctions in the node (those MAs which are not MAPs). However, once this is solved the implementation is easy again, since there is no need to configure the network itself and implement the protocol in a static way.

Note that in all cases the Logical Network is going to be different for each and every Mobile Node.

Now let's give a simple method to choose the MAs that will be used to construct the hierarchy tree of the network: At the beginning the MN is attached to its HA, then it moves to another MAP. The MN records all the MAs along the way (from the MAP to HA). Then when it makes a handover it records the way again. The first common element of the route (from the MN) is then dedicated to be a Hierarchy Point.

This method is very easy to implement and rather simple. We show in our simulation work that it still outperforms the basic protocols.

4.1.3. Personal tracking mobile IP – PTMIP

A tracking-like (see Fig. 2) solution would be again easy to implement. In this case the *tracking handover* is introduced when the MN orders the new MAP to report always only to the previous MAP it

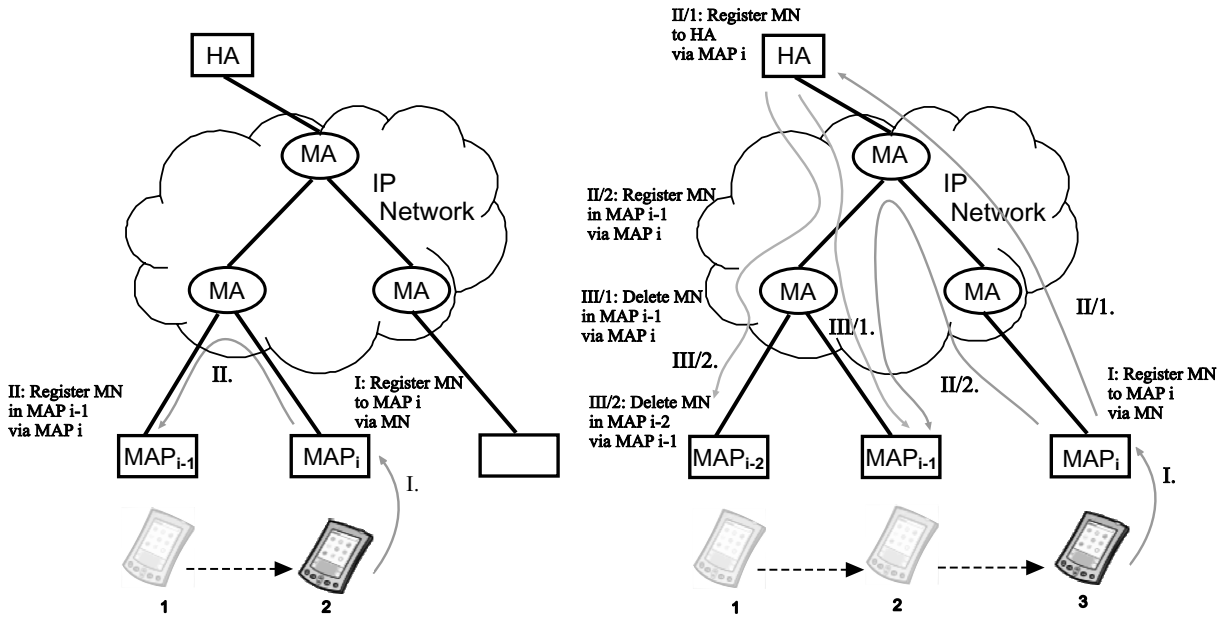

Fig. 2. The operation of the Personal Tracking Mobile IP protocol. The *tracking handover* is depicted on the left hand side while the right is about the *normal handover*.

was attached to, like in the DHMIP [26] or LTRACK [5] protocols.

The following message structure would be used for the *tracking handover*.

[Dst: MAP$_i$, Src: MN, Actions: Register MN to MAP$_i$ via MN;
[Dst: MAP$_{i-1}$ //The former node//,
Src: MN, Actions: Register MN to MAP$_{i-1}$ via MAP$_i$
]
]

When the MN is paged, the message is sent through all the nodes along the way. For this reason, after a number of *tracking handovers* the MN performs a *normal handover*, i.e. registers back to the HA (or to some hierarchy point in a more complex solution). One possible implementation of the *normal handover* would look like the following.

[Dst: MAP$_i$, Src: MN, Actions: Register MN to MAP$_i$ via MN;
]
[Dst: MAP$_{i-1}$ //The former node//,
Src: MN, Actions: Register MN to MAP$_{i-1}$ via MAP$_i$
]

There are many proposed methods to decide between the two type of handovers. In our simulation we implemented a simple suboptimal solution when the MN registers back at every *i*th step. We analytically compare the results with the one that uses statistical handover decision and with the optimized LTRACK [5].

4.1.4. Personal cellular mobil IP – PCMIP

Since the widespread use in GSM the cellular solutions became popular in most mobility applications. The idea is to avoid registrations when the MN moves within a given set of MAPs but then search it at each when it is paged. There is a great literature cell forming algorithms. We give an alternative one.

We want to point out that in this case the paging areas are different for each MN and are formed in an almost optimal way by each MN individually. We expect better performance in large networks.

The MN should send registration messages only when it moves to a new Paging Range (PR). In this case it orders the leader of the new Paging Range to register at an upper level that the MN is in the PR. Also the MN tells the IDs of the MAPs in the Paging Range to the leader of the PR so it will be aware who to broadcast the messages when the mobile is paged.

The following message tells to the one specific $\text{MAP}_{\text{leader}}$ (the leader) the MAPs belonging to that given PR: $\text{MAP}_i, \text{MAP}_{ii}, \ldots$:

[Dst: $\text{MAP}_{\text{leader}}$ //The leader of the paging area//,
Src: MN, Actions: Register MN to $\text{MAP}_{\text{leader}}$ via MAP_i, $\text{MAP}_{ii}, \ldots$,
[Dst: HA, Src: $\text{MAP}_{\text{leader}}$, Actions: Register MN to HA via $\text{MAP}_{\text{leader}}$
]
]

And a message that can be used to register the PR of the MN at an upper level:

[Dst: MAP_i, Src: MN, Actions: Register MN to MAP_i via MN;
]

The problem to solve for cellular algorithms is the problem of forming he Paging Ranges. Forming the cells at an optimal cost using the total frequency of handovers on aggregate level (not individually for each MN) is NP hard. Consecutively the problem is NP hard for only one MN too. However, there are alternative solutions giving a solution what is good enough.

4.2. Determine the optimal mobility management strategy

The MN node continuously builds the previously mentioned logical network, consisting of node and edges with different weights, with information gathered from the visited network segment. Based on the logical network stored as an adjacency matrix and the derived network parameters the mobile station always has to be able to determine the most optimal mobility management strategy for itself. MN maintains the following matrix and vector besides the visited logical network:

$$Q_{change} = \begin{vmatrix} \infty & C_{C->H} & C_{C->T} & C_{C->Ce} \\ C_{H->C} & \infty & C_{H->T} & C_{H->Ce} \\ C_{T->C} & C_{T->H} & \infty & C_{T->Ce} \\ C_{Ce->C} & C_{Ce->H} & C_{Ce->T} & \infty \end{vmatrix}.$$

$$Q_{cost} = \left| C_C \; C_H \; C_T \; C_{Ce} \right|.$$

Where C, H, T, Ce is the abbreviation of centralized, hierarchical, tracking and cellular strategies, and $Ctech1->tech2$ is the cost of switching from *tech1* to *tech2*, C_{tech} is the summarized, possible cost of the tech management strategy, which can be seen in details in one of our previous work [16,18]. This matrix and vector can certainly be greater based depending on the number of mobility management strategies known by the mobile.

During its movement if MN reaches a new node or and old one, but updates the Q_{LN} matrix with new parameters, the $Q_{changecost}$ matrix and Q_{cost} vector is updated accordingly. Then it checks if it is worth to change from the actual management strategy ($tech_{actual}$) or not:

$$C_{tech_{actual}} < C_{tech_i} + C_{tech_{actual}->tech_i}, \forall i \tag{1}$$

If the mobile finds a strategy which it should switch to, then it performs the technological handover minimizing and always providing the most optimal mobility support for itself and the network.

5. Algorithms to set the parameters of the above protocols

At first we briefly present what kind of information can the MN simply record to have a more sophisticated view of the Logical Network.

Then we show algorithms for the protocols presented above that can optimize their performance.

5.1. Collecting logical network data

The algorithms we propose require a set of extra data to be recorded by the MN. After switching on, the MN collects data from every network it attached. Let $N_t = \max K, N$ denote the number of networks visited at time t.

The parameters we suggest to record:

– u_i: the cost of update from node i to the HA; can be determined from the traceroute times
– $u_{i,j}$: the cost of update from i to j; can be determined from the traceroute times
– d_i: the cost of delivery from the HA when the MN is paged;detto
– $d_{i,j}$: the cost of delivery from node i to node j;detto
– μ_i: the number of calls received at node i;
– $\lambda_{i,j}$: the number of movements from node i to node j,
– λ_j: the number of movements to node j.

These parameters will be used to model the network from the MN point of view, to decide the strategy and the actions. (This network model is very similar to the one used in our Mobility Management Framework (MMF) [16–18].)

One can extend the approach with recording for example Quality of Service (QoS) data or making reliability measurements, calculating costs of using each MAP (costs might be different if the IP mobility uses different service providers). However, in the present work we disregard these factors.

5.2. The optimal number of tracking handovers for PTMIP

The MN itself should decide whether to do *tracking handover* or *normal handover*. The optimal number of tracking handovers for the classical LTRACK solution using a Markov Chain model is calculated in [5]. Here we give a more simple alternative solution that gives an unique result for each MN depending also on their exact location and then compare it with the global solution.

5.2.1. Cost expectation minimization algorithm

Assume that the MN has just attached to a new MAP and has to decide upon the handovers. We assume that the best guess of the parameters of the consecutive MAP the MN will attach to is the same as the parameters of the actual MAP. (These assumptions is very much like the martingale property assumption for stock value changes or call frequency changes in communication networks. To further underpin it we should make measurements on real networks.) Furthermore, if the parameters of the actual MAP are not known than it is assumed to be the same as the previous etc. (If only the MN moves to a MAP for which it have measurements from the past then of course it can use those for the calculation.)

It is worth to make a *normal handover* only if its expected cost is less than the expected cost of the tracking handover. All the cost have to be calculated considering possible deliveries too.

The expected cost with the *tracking handover* (at node i):

$$u_{i,\text{prev}} + \frac{\mu_i}{\mu_i + \mu_j} D_i,$$

where $D_i = d_1 + d_{1,2} + \ldots + d_{\text{prev},i}$ is the total cost of delivery through the chain of tracking points. The expected cost of a *normal handover*:

$$u_i + \frac{\mu_i}{\mu_i + \mu_j} d_i.$$

After the costs recorded the decision is easy to make.

5.2.2. Comparing PTMIP to LTRACK

We compare the original LTRACK solution, what is now referred as the most efficient version of the classical mobility approaches, to the PTMIP solution with cost expectation minimization handover strategy.

Suppose the operator implemented LTRACK to the network with optimized parameters. This means that smart MNs are moving from MAP to MAP making handover decisions based on measurements they make.

Let us examine one handover generally supposing that it is not a loop (for loops the costs are the same for both protocols). We are at position MAP' and we move to MAP". Then the update and delivery cost from these two access points to the HA are $u', u", d', d"$ respectively. The cost of a tracking handover is supposed to be u_t and the corresponding delivery is d_t. The experimental probability of receiving a call at MAP" is p.

We can suppose that the optimal number of tracking handovers for LTRACK is $r/(1-r)$ so r is implicitly defined as the relative frequency of tracking handovers. The expected cost of the LTRACK protocol is then the following:

$$K_{\text{LT}} := r(u_t + p(d + d_t + u)) + (1 - r)(u + pd). \tag{2}$$

In the PTMIP strategy the MN calculates the expected cost of the tracing and the normal handover and takes the minimum:

$$K_{\text{PT}} := \min K_{\text{T}} := u_t + p(d + d_t + u), K_{\text{N}} := u + pd. \tag{3}$$

The cost differences depend only on the update cost (u) and does not depend on the delivery cost (d). The cost functions as functions of u are depicted on Fig. 3.

In the PTMIP solution we make our decisions on the assumption that $u" = u'$ thus we can decide what is the minimal cost action. Suppose that we are at u' so the MN makes a tracking handover (still without knowing $u"$). The cost of PTMIP is higher than the cost of LTRACK only if $u" < u*$ and smaller in all other cases. (The figure depicts the whole argument on the cost differences.)

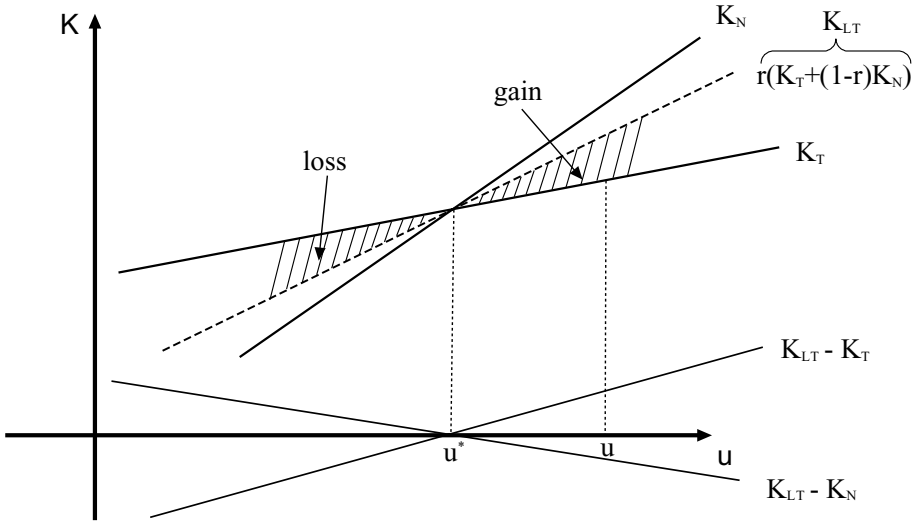

Fig. 3. The cost function of u.

5.3. Cell forming algorithm for PCMIP

We give an algorithm where, based upon a quasi-optimal location area forming algorithm (the problem of optimal cell forming is NP hard), the MN will configure the network to a spacial CIP or TRACKING-like model [16].

With this method the MN is able to operate with almost the optimal cost PR-s and signalling messages are saved too. The tradeoff is that the MN has to maintain its database, calculate the ideal network for itself and also the leaders of the PAs have to maintain a database of the attached mobiles. This shouldn't be too much since different MNs might chose different PR leaders and thus the database and the work is well distributed.

When the MN attaches to a node that is not in its database yet it has to put the new MAP into a PR immediately and notify the leader or even can off-line reconfigure the network.

The solution briefly is the following. The update and registration costs are recorded in each mobile individually for each network node. Then it is decided whether it is worth to merge a cell to another and form a new Paging Range. From now on let us use the word cell what is understood as a common word for node (MA) or Paging Range and also the atomic logical entity of the algorithm.

At first we define the specific cost cost of maintaining a cell, namely: i.

$$c_i = \mu_i d_i + \sum_{\forall j} \lambda_{j,i} u_i = \mu_i d_i + \lambda_i u_i$$

Now we define the extra cost of incorporating a cell into another:

$$c_{i,j} = \mu_j (d_i + d_{i,j}) + (\lambda_j - \lambda_{i,j})(u_{j,i} + u_i) - (\lambda_i - \lambda_{j,i}) u_i$$

Speaking in words, the extra cost when the cell j is incorporated to cell i is the incoming call rate (μ_j) multiplied by the cost of the new delivery procedure through the node i: $(d_i + d_{i,j})$, plus the cost of the updates when the new cell is reached via the incorporated cell j: $(u_{i,j} + u_i)$, multiplied with this event's

frequency: $(\lambda_j - \lambda_{i,j})$, where $\lambda_j := \sum_{\forall k} \lambda_{k,j}$. On the other hand, we also save the cost of moving into cell i from node j that should be multiplied by the frequency of this event: $(\lambda_{j,i})$.

The cost is defined if we want to merge only cell j. Similarly we can define it in the case if we want to merge an $\mathcal{M}$ subset of all the cells:

$$c_{i,\mathcal{M}} = \sum_{m \in \mathcal{M}} \mu_m (d_i + d_{i,m}) + \sum_{n \in N} \sum m \in \mathcal{M} \lambda_{n,m} (u_{m,i} + u_i) \tag{4}$$

Where $\mathcal{M}$ is the set of nodes we want to merge and N is the set of the not merged ones. Also $M \cup N$ is the set of all the nodes examined (neighbors). The algorithm goes as follows: Initially we start from a sorted list of costs of maintaining a cell, and try to merge the neighboring ones with the following rule:

1. Take the first element (node or cell with the maximal c: i);
2. For all M subsets of its neighbors $M \cup N$ (including the trivial ones):
   (a) Take the subset of the maximal cost of incorporation.
   (b) If $\sum_{j \in J} \lambda_j u_j + \mu_j d_j < \sum_{j \in J} c_{i,j}$ i.e. the cost of merging is higher than the cost of maintaining separate cells THEN "DO NOT MERGE" ELSE "MERGE" and recalculate the cost of the new cell. At first some notations and temporary variables:
      - $m \in M$: neighbors Merged + initial node; all the nodes in the new cell
      - $n \in N$: neighbors of the New entity, or all the nodes which were not merged and are not in the new cell
      - $p_m = \frac{\sum_{n \in N} \lambda_{n,m}}{\sum_{nin N} \sum_{l \in M} \lambda_{n,l}}$:the probability that we arrived to the merged node m from outside of the new cell.
      - $\overline{p_m} = \frac{\sum_{n \in N} \lambda_{m,n}}{\sum_{l \in M} \sum_{n \in N} \lambda_{l,n}}$: the probability that we left the new cell from the merged node m.

      Then recalculating each cost:

      - $c_{new} = \frac{c_i + \sum_{m \in M} c_{i,m}}{c_i + \sum_{m \in M} c_m}$ because the c_i is a relative cost;
      - $\lambda_{new,n} = \lambda_{i,n} + \sum_{m \in M} \lambda_{m,n}$;
      - $\lambda_{n,new} = \lambda_{n,i} + \sum_{m \in M} \lambda_{n,m}$ (the frequencies are simply summed);
      - $\mu_{new} = \mu_i + \sum_{m \in M} \mu_m$ this is trivial: the chance of receiving a call in the new cell is the chance of receiving a call in each subcell of it;
      - $u_{new} = u_i + \sum_{m \in M} \sum_{n \in N} \frac{\lambda_{n,m}}{\sum_{n \in N} \lambda_{n,m}} (u_{m,i} + u_i)$ here we should see, that the update cost in the new, "supercell" is the update cost from cell i plus the update costs from the merged cells, what to which subcell we attach to. So we should sum up for every merged neighbor $\sum_{m \in M}$ and then for all the not merged neighbors of each $\sum_{n \in N} \lambda_{n,m}(u_{m,i} + u_i)$;
      - $u_{n,new} = \sum_{m \in M} \frac{\lambda_{n,m}}{\sum_{n \in N} \lambda_{n,m}} u_{n,m}$ We sum up the $u_{n,m}$s, but each has to be normalized with the probability of using it;
      - $d_{new} = d_i + \sum_{m \in M} \mu_m (d_i + d_{i,m})$ The fact is that this works only when one of its 1-distance neighbors are merged. If multiple one is merged then the;
      - $d_{new,n} = \sum_{m \in M} \overline{p_m} d_{m,n}$ what is the weighted value of the delivery cost i.e. sum for all m the cost of delivery from m multiplied by its probability.

   Then sort the new cells (insert the new cell into the right position) and start it again.

The algorithm ends when there is no benefit of merging any cell with any other.

Note that there are two things why this is not the optimal solution. The first is because the leader of the PR is selected without any cost check and secondly because we propose to examine only the neighbors of a PR. (It is mathematically possible that it does not worth to incorporate a neighboring while it worths to incorporate another, far away cell.) However, we still gave a solution good enough.

6. Implementation of the protocol

In this section we discuss the implementation issues of the Client-based Mobility Frame System over IPv4 and IPv6.

During the definition of IPv4 protocol the mobility of the information was not a requirement against the network protocol. Due to this, and the fact that the IP address is used as both global and location identifier, IPv4 does not support mobility as a built-in function. As the mobile moves, it will be assigned a new and different IP address, a so called care of address whenever switching between different sub-networks. However this stands in the way of continuous communication, the higher level protocols are unable to maintain their open connections. As already mentioned a lot of solutions were developed for IPv4. These and IPV4 itself does not offer the possibility to implement our framework, that it the reason why we developed the CMFS protocol. Details are presented in the next chapter.

6.1. CMFS protocol over UDP/IPv4

Let us show a possible CMFS implementation for specification of the above mentioned command structure, into an IPv4 network in the application layer. A CMFS message is carried in UDP packets thus uses the transport and the services from underlaying layers. We have chosen it instead of the TCP, like in other mobility solutions, because the TCP does not operates well with the radio interfaces. The TCP conceives the high bit error rate of the radio channel as congestion, and decrease the window size that ends in significant speed fall-off. For this reason the mobility applications generally use UDP to the communication.

The CMFS message structure follows strict rules as you can see on Fig. 4. The header contains 4 fields of 1 byte elements: type, length, flags, number of actions, two 4 byte elements, an identification section and a variable length actions/informations field. Two different type of CMFS messages are differentiated, a *CMFS request* and a *CMFS reply* message. The length shows the full length of the CMFS packet included the header. Most of the bits of flag field helps the Network Discovery Procedure. One by one the bits mean the following:

- F – The first MA receiving this message sends a *CMFS Reply* and then set this bit to 0.
- S – If the message destination is the given MA and the S=1 then it sends a *CMFS Reply* (partial network discovery)
- L – For any message destination if this bit is set then the MA should send a *CMFS Reply* even though it only forwards the packet (full network discovery)
- C – This bit is examined if the node has to send a *CMFS Reply*. If this bit is set to 1 the MN expect capacity, and overload information from the MA in the *CMFS Reply* message.

The *number of action* field shows the amount of Action/Information contained in the packet payload. If the *destination* field matches the node identifier then it has to process the message and execute the actions in it. The *mobile current IP address* is needed for the Network Discovery Procedure so that

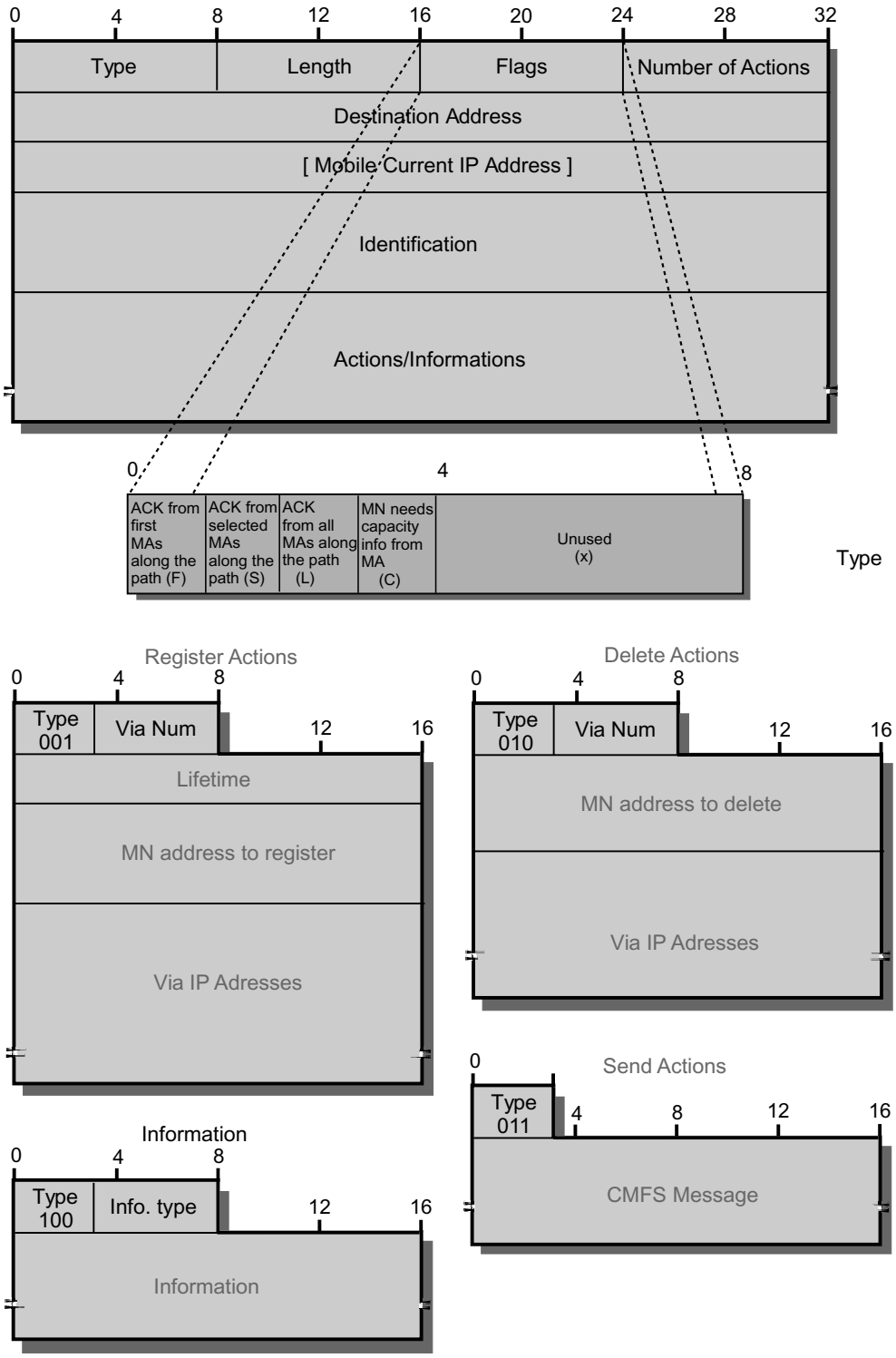

Fig. 4. The CMFS message format.

the MA knows where to send the *CMFS reply* message. For this reason this is a optional field. The *identification* is a 64-bit number that uniquely identifies the CMFS message and is used to match *CMFS requests* to *CMFS replies* and can explicitly provide protection against replay attacks. The *payload* of the CMFS message contains either the actions which have to be accomplished at the specific nodes or useful information, like capacity, for the network entities.

There are three different kind of actions. The first is the Register action that indicates a route registration in the given MA *via* the given destination *to* a specified target. The *lifetime* set by the mobile can assure the soft state data store at the MAs. Cleaning up the IP network is the problem of many mobility management systems. The obsolete registrations in the network nodes cause performance loss or even mis-routing of packets. Because of these it is very important to choose a proper *lifetime* for registrations. An optimal value can save the network from the unnecessary signaling also. The second type is the *Delete* action that erases the specified registered data from the MA database. The *Send* action type instructs the MA, that the payload of the action field has to be send as a CMFS message.

It is essential to have many MAP and MA nodes (i.e. nodes on which CMFS is implement) in the network. The most important is that the protocol stack of the gateways and routers is updated with the CMFS functionality. It is not our scope now to perform exhaustive research to select the optimal set of nodes that should implement CMFS to provide a cost-effective solution.

However, a distributed logical mobility network could be constructed with implementing CMFS on user PCs. Of course the solution would be far from optimal regarding performance issues but tests and measurements could be executed.

6.2. CMFS methodology with mobileIPv6

Internet Protocol version 6 was designed to eliminate limitations and defects of IPv4. Most of these corrections are realized in extensions of the protocol, like the 128 bit network address, flow label, built-in security system, built-in multicast and anycast addresses and what is the most important for as: basics of supporting mobility.

We do not focus on giving a detailed description about Mobile IPv6 developed by the Wireless/Mobile Hosts of IETF [8]. Here we aim to show how to extend Mobile IPv6 to implement the CMFS functionality into it with minimal changes. However, this requires some introduction into Mobile IPv6.

On guiding star when Mobile IPv6 was developed was to keep the IP transparency including to sustain active TCP and UDP connections. Although the MN gets a new Care-of-Address every time it handovers to a different network and on network layer, the MN can be reached on this address and can keep its global address constant. This way MIPv6 allows the MN to change MAP without the change of global IP address that higher layer applications use. This solves the fundamental problem of IPv4 mobility. The MIPv6 allows that not only the Home Agent, but other agents in the network store the care of address linked with the global address thus the triangle dataflow problem of MIPv4 is resolved too.

The administration is managed with the Binding Update (BU), Binding Acknowledgement (BA) and Binding Request (BR) messages. The information about the MNs in the MAs is stored in the so-called binding cache. On the mobile side the list of MAs, where there is information about the location of the MN, are maintained in the so-called binding list.

Binding Update is sent as an IPv6 Destination option, the mobile node may include it in any existing packet (such as a TCP packet) that it is sending to that destination node, or it may send the Binding Update in a packet by itself. Packets which carry a Binding Update must also include an IPv6 Authentication header, which provides authentication and replay protection for the Binding Update.

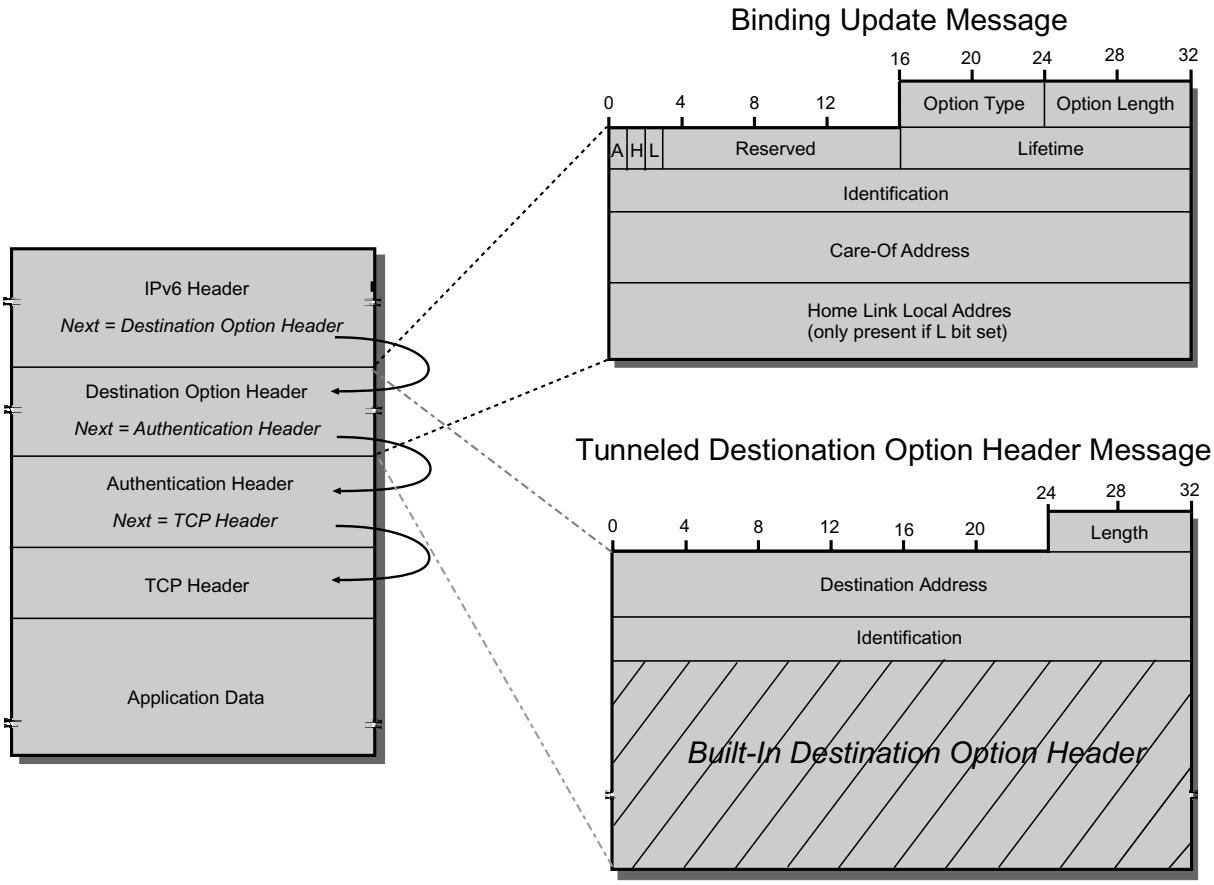

Fig. 5. The IPV6 message structure extension.

Two possible realizations of CMFS on Mobile IPv6:

1: The CMFS based mobility management strategies can be implemented using the Binding Updates messages of MIPv6 in the appropriate sequence with the appropriate parametrization. There is no need to extend the existing functionalities of the protocol. The backwards of such a solution is that the packet forwarding function described in chapter 3.3.2 can not be implemented: the MN has to send all the messages. This would result in the following modification of the PMIP messages:

[Dst: MAP_i, Src: MN, Actions: Register MN to MAP_i via MN];
[Dst: HA, Src: MN, Actions: Register MN to HA via MAP_i];
[Dst: MAP_{i-1}, Src: MN, Actions: Delete MN in MAP_{i-1} via MN].

In order to minimize the chance of the out of sync event, that is a message sent later arrives earlier, the MN can use the routing header to lead the message sequentially to those MAs it would have reached in the original solution.

2: Tunneled Destination Option – In order to preserve the CMFS packet forwarding function and to provide all of the previously detailed services a Tunneled Destination Option extension was developed (Fig. 5). The basic idea is that the agent IPv6 node has to interpret a so called packed Destination Option message. This contains a MobileIPv6 message, e.g. BU, or another Tunneled Destination Option. If

an agent station receives this message, it unpacks it. In case the message is TDOH the station will forward it to the given Destination, if a BU, than to the destination given in the BU. As source address the agent indicates its own address. The Identification field is used to ensure that Binding Updates are not applied out of order at the destination in spite of varying network delays, and to match Binding Acknowledgements with outstanding Binding Updates at the mobile node. The advantage of this solution is that the whole CMFS function can be preserved; the disadvantage is that the nodes have to be updated to be able to interpret the Tunneled Destination Option message.

Both solutions could work well, but we suggest using the second option to provide QoS. The IPv6 Destination Option Extension message sending mechanism will be demonstrated in the following examples using previously introduced mobility management strategies.

The Destination Option Header in PMIP over MobileIPv6 is the following:

BU{Src: CurrentCoA, Dst: MAP_i, HomeAddress:MAP_i};
TDOH{Payload:[BU{Src: MAP_i, Dst: HA, HomeAddress:MNHome, CoA:MAP_i},
BU{Src: MAP_i, Dst: Src: MAP_{i-1}, HomeAddress:MAP_{i-1},
Lifetime:0}
]
}.

where TDOH means Tunneled Destination Option Header, CurrentCoA is the mobile node's current IP address, and MNHome is the mobile node's home address.

At first we define a Binding Update in the structure, with which we refresh our binding at the actual MAP. We also define a Tunneled Destination Option Header, which contains two BUs. Both will be forwarded by MAP_i. The first will be forwarded to HA, the second to MAP_{i-1}. Lifetime $= 0$ in the second BU indicates that the record corresponding to the mobile should be deleted.

The Destination Option Header with Extended PMIP over MobileIPv6 is the following:

BU{Src: CurrentCoA, Dst: MAP_i, HomeAddress:MAP_i};
TDOH{Dst: MAP_{i-1},
Payload:[BU{Src: MAP_i, Dst: MAP_{i-1},
HomeAddress:MAP_{i-1}, CoA:MAP_i},
TDOH{Dst: HA,
Payload: [BU{Src: MAP_{i-1}, Dst: HA,
HomeAddress:MNHome, CoA:MAP_{i-1}},
TDOH{Payload: [BU{Src: HA, Dst: MAP_{i-1},
HomeAddress:MAP_{i-1}, Lifetime=0}]
]
}
]
}.

This extended PMIP example can be interpreted similar to a PMIP. The reason if Destination is not given in a TDOH is that it only contains a BU with a given Dst address. It can clearly be seen that there will be multiple tunneling in and out, and complicated message structure, which can be omitted; they are only given in these examples to compare to the examples above and to demonstrate the operation.

Definition of the most optimal message structure results in the following. E-PMIP functionality can also be reached with this, as with the previous one, but with a lower cost:

BU{Src: CurrentCoA, Dst: MAP_{i-1}, HomeAddress:PrevCoA, CoA: CurrentCoA};
BU{Src: CurrentCoA, Dst: HA, HomeAddress:MNHome};
TDOH{Dst: MAP_{i-1},
Payload:[BU{Src: HA,Dst: MAP_{i-1},
HomeAddress:PrevCoA, Lifetime=0}].}

where PrevCoA is the previous care of address of the mobile node.
The case of the PTMIP is a bit trickier. Let's omit the previously shown (in chapter 4.1) example, and see the most optimal structure:

BU{Src: MNCurrentIP, Dst: MAP$_{i-1}$, HomeAddress:PrevCoA, CoA:MNCurrentIP};

As it can be seen that a simple BU is going out on the line, which is a request towards the agent visited in the previous domain to treat the previous CoA as home address in the future and forward any arriving packets to the actual address of the MN.

To demonstrate the example of CPMIP we use the multicast addressing as a built-in feature of IPv6.

MultiIP:{MAP$_i$,MAP$_j$,MAP$_k$}

BU{Src:CurrentCoA, Dst: MAP$_{leader}$, HomeAddress:MAP$_{leader}$, CoA:MultiIP};
TDOH{Payload:[BU{Src: MAP$_{leader}$, Dst: HA,
HomeAddress:MNHome, CoA:MAP$_{leader}$}]}.

where MultiIP is the IP address set of this addresses :{MAP$_i$,MAP$_j$,MAP$_k$}. Multicast addressing has an important role in that case. In MobilIPv6 it is not possible to define multiple bindings to a mobile, like in IPv4 with CMFSP. However the possibility arises if a mobile can be reached at more addresses, it should be assigned a multicast address meaning that if we send the packets to the mobile with flood manner to each address that is assigned to the group corresponding to the multicast address.

In this chapter we presented how the solutions developed in MobilIPv6 can be used from the CMFS point of view.

7. Simulation

We have made a simulation to show at first that our proposal actually works and secondly to compare it to existing technologies. The simulation was written in the open source OMNet++ [15] using C++ language.

It is essential to point out that the simulation is written such a way that it can easily cooperate with the one presented by us for classical mobility solutions like MIP, CIP, HMIP, LTRACK, etc. We use this to compare the methods. For details see. [16].

7.1. The structure of the CMFS simulation program

The simulation consists of two main modules namely *MN* and *MA* and some other simple components that are needed to model the operation environment (see The two main modules has similar internal structure. Both has a *DataSender* and a *DataReceiver* to be able to send and receive messages while their logic is hidden in *NodeCore_MN* and *NodeCore_MA* respectively.

The whole CMFS protocol is implemented in the *NodeCore* components. The *NodeCore_MN* constructs CMFS messages, maintains a database and builds up the Logical Network. The *NodeCore_MA* understands the CMFS messages and executes the actions, maintains the database and routes the messages and packets using it.

The *DataSender* module creates traffic in the network to a random target and at random times while the *DataReceiver* is responsible to receive and analyze it. The number and size of packets, the frequency of data sending and the possible targets for a node can be set as a parameter of the simulation. The

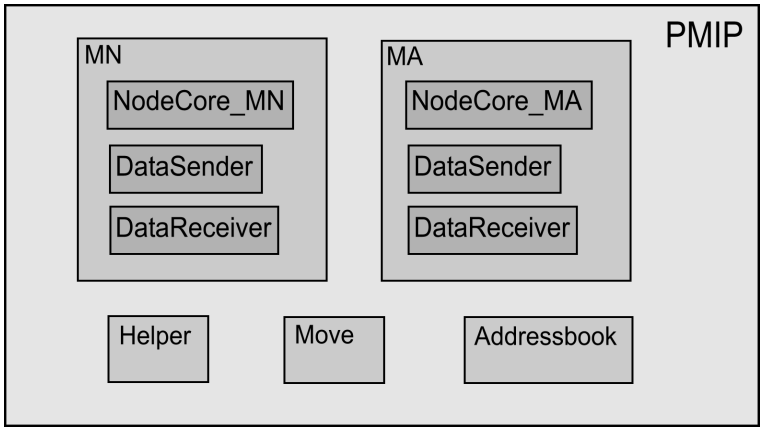

Fig. 6. The component structure of the simulation of CMFS written in OMNet++.

receiving side measures the average number of handovers, number of arrived/sent/lost packets and their averages in 1 min interval but also can be extended to record other QoS parameters like delay or jitter too.

The *Addressbook* module is the template for the databases in the MAs. The module *Move* is responsible for the movement directions and movement frequency of the MNs. The *Helper* component implements some functions and objects that are not logically part of any above.

7.2. The tested scenario

We have constructed a virtual test environment consisting of 9 MAs and 9 MNs with the initial MN distribution depicted on Fig. 7.

8. Comparison and analysis of the protocols and some numerical results

In this section we grabbed three main aspects of the new solution to analyze it. At first we show how its costs relates to the simple MIPv4 solution purely collecting network signalling data and examining the whole from QoS point of view. Then we give some numerical results on the performance of a more complex solution, PTMIP. Finally we discuss a few conceptional issues that reveal some fundamental differences between our new protocol and the MIP versus its existing extensions.

8.1. Comparing the basic approaches

We have run the simulation on various mobility parameters for all the algorithms separately. All the nodes make calls according to a Poisson process to random targets with a biased uniform distribution so about 80% of the calls are terminated at mobile clients. The mobility ratio (number of handovers per received call) is varied to show how it affects the performance.

The performance of the protocols is depicted on Fig. 8. However at low mobility level (when there are only a few handovers between two calls) E-PMIP is better than the classical MIPv4 but as the mobility ratio rises the protocol performs worse in terms of signalling load on the network. It is because it requires more operations and messages in the network to provide the better QoS parameters.

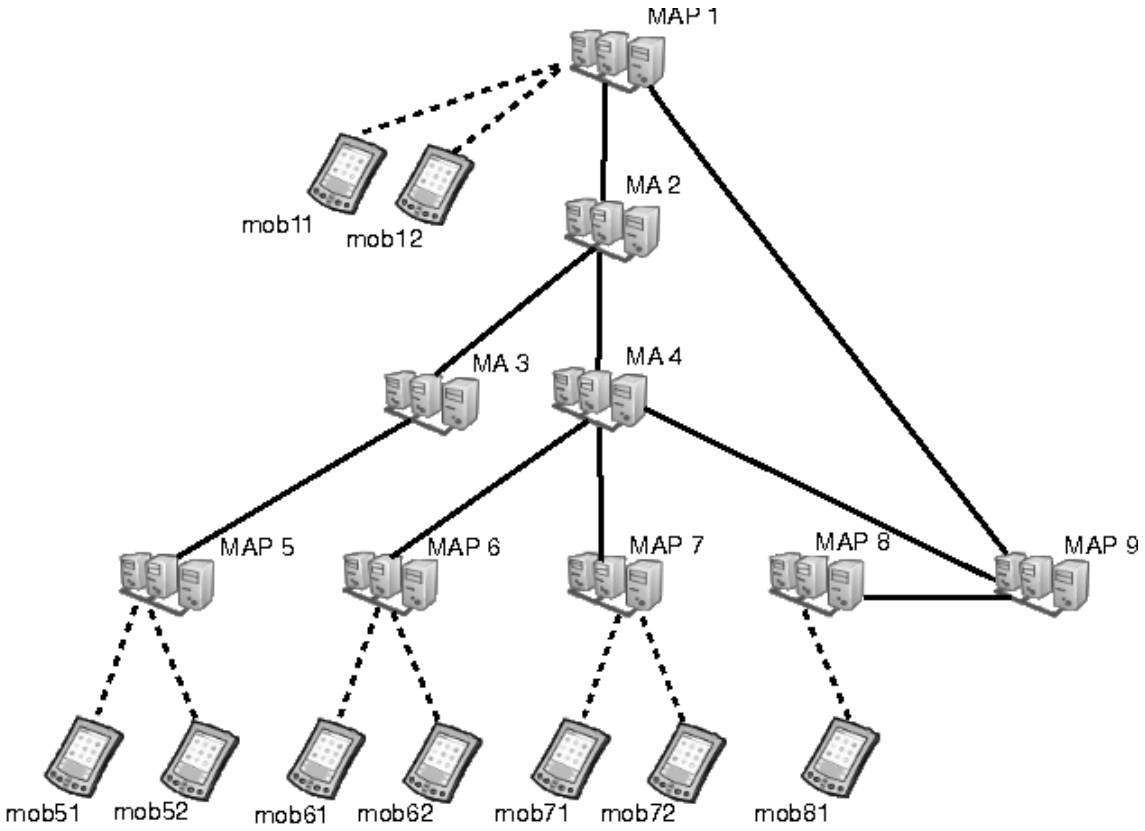

Fig. 7. The test network in OMNet++.

We can see that the P-PMIP is always better than the MIPv4. This is because if we look at the two protocols both have the same signalling strategy but MIPv4 need *Agent Advertisement* messages to maintain connectivity while in the client based system can rely on lower layers.

What we can conclude is that the basic solutions work with approximately the same cost. However, E-PMIP shows that it is possible to improve the performance while not changing the protocol at all (only on the MN side).

8.1.1. Comparison with a QoS-like cost function

We have already presented a figure to compare the approaches from the signalling load point of view. Of course it is easy to define a kind of QoS cost function too and examine the performance in the mirror of that. We construct a very simple kind of QoS cost function.

We can assume that the *MN can attach to only one MAP at a time*. If not than both protocols could equally benefit from the fact that the MN can receive duplicated messages while changing the MAP. The second assumption is that the *signalling and the data is transmitted with the same speed between MAs*. This is rather reasonable or at least if there is a difference because of IP packet prioritization then the same method is expected to be applied for the classical and the new solutions. As a third assumption we say that because of the second one, *the transmission speed is in the same linear relation to the logical distance of MAs for both solutions*. We introduce a network and mobility related parameter g_T that is the distance from the previous MA the MN has attached and the new one. Let us note the average

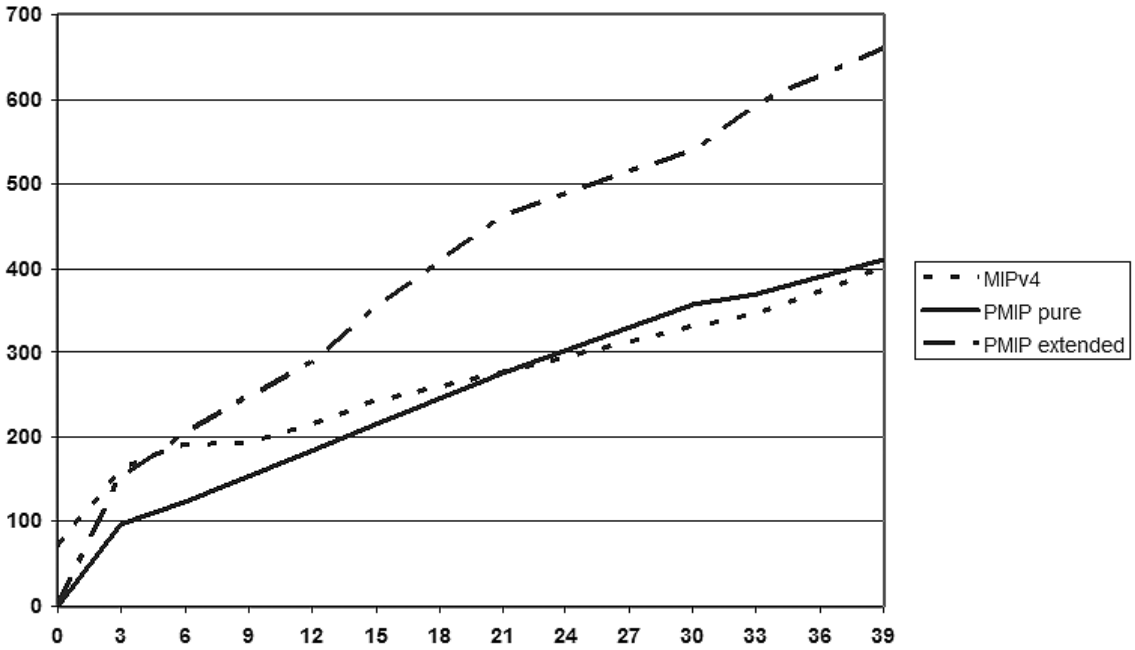

Fig. 8. Comparison of the three mobility Management Systems presented. The horizontal axis shows the number of handovers between two arrived calls while the number of bytes transmitted on the network by each protocol is presented on the vertical axis.

distance from the HA with m. Similarly g_C will denote the logical distance to the FA or MAP the MN attaches. The $t^{\text{Protocol}}_{\text{MA}}$ and $t^{\text{Protocol}}_{\text{HA}}$ will denote the average time of processing Protocol messages at the specified nodes.

We measure QoS cost in the relative number of packets to wrong direction for each protocol. We assume that this is a linear function of time thus a linear function of the logical distance and processing time. This linear function can be set to the same value for all the protocols and will be chosen to Identity for simplicity. (Of course this comparison can be further elaborated if intended.) Now let us see the QoS cost:

$$\begin{aligned} C^Q_{\text{MIP}} &= g_C + t^{\text{MIP}}_{\text{MA}} + mt^{\text{MIP}}_{\text{HA}} \\ C^Q_{\text{P–PMIP}} &= g_C + t^{\text{P–PMIP}}_{\text{MA}} + mt^{\text{P–PMIP}}_{\text{HA}} \end{aligned} \tag{5}$$

$$C^Q_{\text{E-PMIP}} = g_C + t^{\text{E-PMIP}}_{\text{MA}} + g_T t^{\text{E-PMIP}}_{\text{MA}} \tag{6}$$

and thus we can derive the QoS cost relation between the protocols:

$$C^Q_{\text{MIP}} - C^Q_{\text{P–PMIP}} = (t^{\text{MIP}}_{\text{MA}} + t^{\text{MIP}}_{\text{HA}}) - (t^{\text{P–PMIP}}_{\text{MA}} + t^{\text{P–PMIP}}_{\text{HA}}), \tag{7}$$

that is rather hard to handle, since depends very much on the implementation and the working nodes. But it is easier to see the clear difference between the MIP and E-PMIP if we assume that all processing times are equal to t_{PROC}:

$$C^Q_{\text{MIP}} - C^Q_{\text{E-PMIP}} = m - g_T. \tag{8}$$

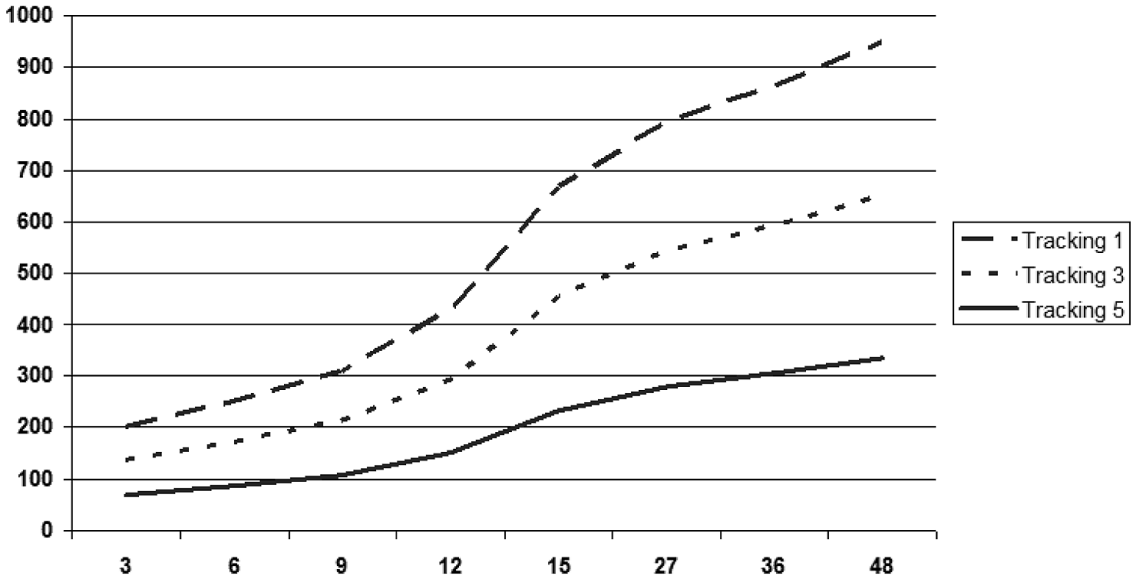

Fig. 9. This figure depicts the perfromance of three tracking-like approaches namely PTMIP with 1, 3, 5 tracking handovers. Note that for this simulations a couple of additional links were inserted to the network.

This is the time difference in traffic disturbance. It is somehow logical to see that in most of the times $m \geqslant g_T$ since the HA is often farer from the new MA than the old MA.

8.2. Numerical results for complex mobility managements using CMFS

We have run some simulation to provide results on the performance of the tracking-like solution PTMIP. In this case we speak about such a tracking when the messages are sent through the links of the network not through the air interface.

On Fig. 9 one can see that in this case the most *tracking handovers* performed and thus the most *normal handovers* avoided significantly decreases the signalling cost for the protocol.

8.3. Differences in the network implementation

In this Section we try to focus on the benefits of the fact that the client based mobility management does not need a pre-built network topology. Both in the simulation and in the case of the above calculations we had an assumption that the same nodes run mobility management system in the network for the MIPv4 and the PMIP protocols. We say that this is not necessarily true.

If we implement Mobile IP to a system, we have to install HA and FA functionality into all the networks we want to use for the communication. If a new FA is installed to the system it can automatically co-operate with all the different HA in the network according to the RFC. This is the same in our proposal and there is no need for a complex network structure.

The differences appear if we want to extend the algorithms. The classical MIPv4 enhancements such as HMIP [6], TeleMIP [22], DHMIP [26], CMIP [2], Hierarchical Paging [13], LTRACK [5], etc need to have a pre-built network structure i.e. the MAs in a HMIP has to be aware of the hierarchy structure. It is a great advantage that this is not needed in our case since the MN itself can administrate the Logical

Network for itself. This also allows to build different hierarchical or cellular structures for each node in an optimal way to provide QoS or ensure cheap operation (low signalling or processing cost).

One can see that any node can join and leave the network at anytime just like in the MIP case but meanwhile a more complex mobility management can be applied. To discuss the cases when an FA or an MA breaks down or joins the network is part of our upcoming work.

9. Conclusion

We have introduced a mobility management system that solves IP mobility from a very different point of view than any other mechanism before. We have shown example algorithms taking ideas from classical solutions. We prepared a simulation and tested our protocol in operation. Using it we compared the performance of some basic solutions and we have shown that extensions may be beneficial for both the MN and the network.

Further extensions: Since the MN records the details of a MAP it can also perform quality measurement or reliability measurement thus classify the MAPs and networks and use this information in the future (for example when multiple MAPs are available).

We have shown how CMFS would work over IP. However, it is rather simple to extend the whole to IMS too.

Acknowledgements

The research was motivated by the High Speed Network Laboratory and was partially founded by the Hungarian Ministry of Culture and Education with reference numbers NK 63066 and TS 49835 and Mobile Innovation Center. Special thanks to Péter Orbán (Ericsson Telecommunications Hungary) for the careful reviews and support.

References

[1] A. Diab, A. Mitschele-Thiel, *Minimizing Mobile IP Handoff Latency*, 2nd international Conference on Wired/Wireless Internet Communications, 2004.
[2] A.T. Campbell, J. Gomez and A.G. Valkó, An Overview of Cellural IP, *IEEE* (1999), pp. 29–34.
[3] A. Fongen, C. Larsen, G. Ghinea, S.J.E. Taylor and T. Serif, Location based mobile computing – A tuplespace perspective, *Mobile Information Systems* **2**(2–3) (2006), 135–149.
[4] A.K. Pandey and H. Fujinoki, Study of MANET routing protocols by GloMoSim simulator, *International Journal of Network Management*, 2005.
[5] B. Kovács, M. Szalay and S. Imre, Modelling and Quantitative Analysis of LTRACK – A Novel Mobility Management Algorithm, *Mobile Information Systems* **2** (1/2006) (2006), 21–50.
[6] C. Castelluccia, A Hierarchical Mobile IP Proposal, Inria Technical Report, 1998.
[7] C. Castelluccia, *HMIPv6: A Hierarchical Mobile IPv6 Proposal*, ACM SIGMOBILE Mobile Computing and Communications Review (2000), 48–59.
[8] C. Perkins, Mobile IPv6, *Proceedings of the 26th Annual IEEE Conference on Local Computer Networks*, 2000.
[9] C. Perkins, Mobile IP, *IEEE Communications Magazine*, 1997.
[10] C. Abondo and S. Pierre, Dynamic location and forwarding pointers for mobility management, *Mobile Information Systems* **1**(1) (2005), 3–24.
[11] H. Schulzrinne and J. Rosenberg, The Session Initiation Protocol:Internet-Centric Signaling, *IEEE Communications Magazine*, 2000.
[12] J.W. Jung, R. Mudumbai and D. Montgomery, Performance Evaluation of Two Layered Mobility Management using Mobile IP and Session Initiation Protocol, Global Telecommunications Conference (GlobCom'03), 2003.

[13] M. Szalay and S. Imre, Hierarchical Paging- Efficient Location Management, Fourth European Conference on Universal Multiservice Networks (ECUMN'07) (2007), 301–310.
[14] Marcellin Diha and Samuel Pierre, Home Agent Architecture and Algorithms for Mobility Management in Mobile IP Networks, *Journal of Computer Sciences* **2**(1) (2006), 01–06.
[15] OMNet++, http://www.omnetpp.org/.
[16] P. Fülöp, B. Kovács and S. Imre, Study on mobility management modelling methods, *Proc. The 4th International Conference on Advances in Mobile Computing and Multimedia* (*MoMM 2006*) (2006), 17–26.
[17] P. Fülöp, B. Kovács and S. Imre, Numerical analysis of mobility management methods, Info-communications-technology, 1945 volume LXII 2007/7 (2007), 32–38.
[18] P. Fülöp, B. Kovács and S. Imre, Enhanced Mobility Management Modelling Framework, *Proc. 6th Computer Information Systems and Industrial Management Applications* (*CISIM 2007*) (2007), 53–58.
[19] P. Fülöp, B. Kovács and S. Imre, A Client-driven Mobility Frame System – Mobility Management From a New Point of View, *Proc. 13th International Telecommunications Network Strategy and Planning Symposium* (*Networks 2008*), 2008.
[20] P. Jokela, P. Nikander, J. Melen, J. Ylitalo and J. Wall, Host Identity Protocol: Achieving IPv4 – IPv6 handovers without tunneling, *Proceedings of Evolute workshop 2003: Beyond 3G Evolution of Systems and Services*, University of Surrey, Guildford, UK (2003).
[21] R. Ramjee, T. La Porta, S.Thuel, K. Varadhan and L. Salgarelli, A Hierarchical Mobile IP Proposal, Inria Technical Report, 1998.
[22] S. Das, A.Misra, P. Agraval and S.K. Das, TeleMIP: Telecommunications-Enhanced Mobile IP Architecture for Fast Intradomain Mobility, IEEE Personal Communications (2000), 50–58.
[23] V. Simon and S. Imre, A simulated annealing based location area optimization in next generation mobile networks, *Mobile Information Systems* **3**(3–4) (2007), 221–232.
[24] V. Simon and S. Imre, Location Area Design Algorithms for Minimizing Signalling Costs in Mobile Networks, *International Journal of Business Data Communications and Networking* (*IJBDCN*), 2007.
[25] W. Fritsche and F. Heissenhuber, *Mobile IPv6 – Mobility Support for the Next Generation Internet*, IABG mbH, 2000.
[26] W. Ma, and Y. Fang, Dynamic Hierarchical Mobility Management Strategy for Mobile IP Networks, *IEEE Journal of Selected Areas In Communications*, 2004.
[27] Wolfram Research Inc.: *Mathematica*. http://www.wolfram.com/.

Benedek Kovács received his M.Sc. degree in Technical Informatics from the Budapest University of Technology and Economics in 2006. Currently he is a PhD student on the faculty of Mathematics at the same university on the Department of Mathematical Analysis and member if the High Speed Network Laboratory, Hungary. Currently he is doing research work at Ericsson Telecommunications Hungary, Traffic Lab. His research interest is IP Mobility, Overload and Load Regulation methods and intensity estimation of Stochastic processes in telecommunications and parameter estimation of dynamic processes. The topic of the upcoming PhD theses is the optimal control of dynamical stochastic systems.

Peter Fülöp received his M.Sc. degree in Technical Informatics from the Budapest University of Technology and Economics in 2005. Currently he is a PhD student on the faculty of Informatics?at the same university on the Department of Telecommunications. He is working as a system test engineer at Ericsson Telecommunications Hungary at Research and Development Department. His research interest is IP mobility, interworking of heterogeneous mobile networks and movement modeling in cellular, mobile networks. The PhD thesis is going to be written on Complex Mobility Management Systems and Applications.

Sándor Imre was born in Budapest in 1969. He received the M.Sc. degree in Electronic Engineering from the Budapest University of Technology (BUTE) in 1993. Next he started his Ph. D. studies at BUTE and obtained dr. univ. degree in 1996, Ph.D. degree in 1999 and DSc degree in 2007. Currently he is carrying his teaching activities as Head of the Dept. of Telecommunications of BUTE. He was invited to join the Mobile Innovation Centre of BUTE as R&D director in 2005. His research interests include mobile and wireless systems. Especially he has contributions on different wireless access technologies, mobility protocols and reconfigutable systems.