A distributed mobility management scheme in flat mobile architecture

Hua Yang[1] and Naoki Wakamiya[2]
Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka 565-0871, Japan
E-mail : [1]h-yang@ist.osaka-u.ac.jp , [2]wakamiya@ist.osaka-u.ac.jp

Abstract: To accommodate exploding mobile data traffic, considerable number of devices, and heterogeneous applications a lot of researches on mobile core networks are oriented from hierarchical to flat architecture. In this paper, we first propose a novel flat architecture to separate the C-plane mobility management function to two virtual servers, i.e. VDMMEs for mobility management and DMDs for mobility databases. Then, we propose a mechanism and an algorithm based on the response threshold model to dynamically and adaptively allocate a VDMME and a DMD appropriate for an MN. We show our proposal can achieve lower delay and lower C-plane overhead than a comparative method.

Keywords— **distributed mobility management, virtual servers, response threshold model, C-plane**

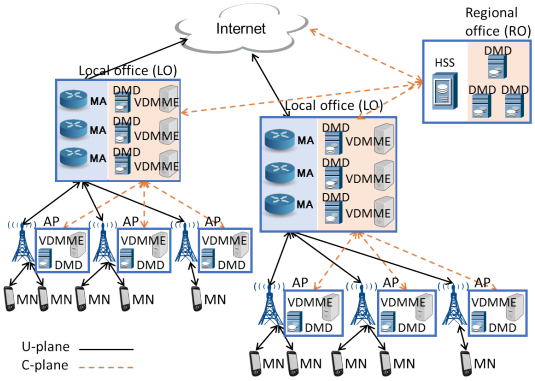

Figure 1. Flat mobile core network architecture

1. Introduction

Recently, with fast development of mobile and wireless communication technologies, mobile and wireless communication networks have been becoming ubiquitous and indispensable in our daily lives, which leads to exploding mobile data traffic, considerable number of devices, and heterogeneous applications. To overcome issues caused by centralized control in the current 3.9G LTE/EPC networks architecture, latest researches are oriented to flat architecture like for example DMM (Distributed Mobility Management) [3][5]. Distribution of mobility management contributes to reduction of delay in U and C-planes and mitigation of C-plane overhead.

In this paper, we propose a novel flat architecture based on DMM [1]. To reduce the response delay and overhead in both of U-plane and C-plane, we separate mobility management tasks from mobility management context data, both of which a mobility management entities (MME) in the conventional architecture manages together. Then we individually distribute them as vertual servers, that is, virtualized and distributed mobility management entities (VDMMEs) and distributed mobility databases (DMDs), over a mobile core network with a help of virtualization technologies. By such separation, we can dynamically and adaptively allocate a VDMME and a DMD appropriate for an MN (mobile node) in accordance with various characteristics of mobile nodes and status of virtual servers. For this purpose, we propose an algorithm and a mechanism of server allocation based on a biologically-inspired response threshold model, which models biological division of labour in social insects for autonomous and distributed decision making [2]. We verify the effectiveness of our proposal through comparison with a DMM-like scheme from viewpoints of delay and C-plane overhead.

The rest of this paper is organized as follow. First, in Section 2, we explain our flat architecture of a mobile core network. Next in Section 3, we give details of our mechanism and algorithm of dynamic, adaptive, and autonomous server allocation. Then we show simulation results and discuss the effectiveness of our proposal in Section 4. Finally we summarize the paper and show future directions in Section 5.

2. Architecture

In our architecture, a mobile core network consists of APs, LOs, and ROs as shown in Fig. 1. A RO has an HSS (Home Subscriber Server) and covers a certain region of areas, more specifically, TAs (Tracking Areas). There are several LOs in a TA and each LO covers several cells. There is one AP per cell. In conventional architecture, delays between an MN and an MME and between an MA (Mobile Anchor) and an MME in C-plane management is not negligible. Therefore, we distribute MAs among LOs. Each MA manages U-plane traffic of an AP under the LO. Furthermore, we separate MME's functions into two parts, the mobility management functions and database to store MN context data, i.e. VDMMEs and DMDs. With a help of virtualization technologies, VDMMEs and DMDs can be dynamically launched and configured at APs, LOs, and ROs appropriate for MNs. We hereafter call VDMMEs and DMDs together servers.

Our architecture seems hierarchical, but the hierarchy is rather based on physical segmentation. There is no master-slave relationship between VDMMEs or between DMDs in different locations. In addition, regarding the U-plane, data packets sent from an MN go to the external network by a corresponding MA via an AP. Thus, a mobile core network has few levels of routing hierarchy in data path. In this sense, we call ours flat architecture as in [1].

For an immobile MN, the nearest AP is the optimal location of servers. On the contrary, when we take the same approach for a mobile MN, reallocation occurs every time an

MN moves between cells. Consequently C-plane overhead caused by migration of MN context between DMDs in APs considerably increases. To mitigate C-plane overhead without sacrificing delay, it is better to allocate a VDMME at the nearest AP and a DMD at the nearest AP or LO, respectively, to an MN with low mobility or low communication frequency. For an MN with high mobility, a VDMME in a LO and a DMD in a RO are preferred. To allocate servers appropriate for each MN taking into account several factors, such as location and mobility characteristics of an MN, delay, bandwidth consumption, communication frequency, load on servers, and C-plane overhead, we propose a mechanism and an algorithm of adaptive and dynamic allocation in the next section.

3. Autonomous allocation of VDMME and DMD

In this section, we first explain a mechanism of dynamic allocation of a VDMME and a DMD. Then we give details of an algorithm to select and allocate appropriate ones to an MN.

3.1 Mechanism

Allocation of servers is initiated by a request message of handover, TAU, or switching sent from an MN. First when an AP of a cell where an MN resides receives a request message, it forwards the request to a VDMME and a DMD which are allocated to serve the requesting MN. We call them a current VDMME and a current DMD, respectively. In addition, the AP forwards the request to VDMMEs in the AP and the nearest LO and to DMDs in the AP, the nearest LO, and the nearest RO. We call those VDMMEs and DMDs, including current ones, which receive the request, member VDMMEs and member DMDs, respectively.

On receiving the request, they autonomously decide whether to serve the requesting MN or not by using our algorithm explained in the next section. Those VDMMEs that decide to become a new current VDMME, which we call candidate VDMMEs, notify the current DMD of their decision. If there are more than one candidate, the current DMD selects a candidate VDMME from which it receives a notification first. Otherwise the current VDMME maintains its service. Similarly, candidate DMDs send notifications to an HSS and an HSS selects one among them as a new current DMD. Results of selection are notified to selected candidates.

A selected VDMME receives MN context from the current DMD. It takes over the role of mobility management from a current VDMME and becomes a new current VDMME after receiving a server assignment response from an HSS with an identifier of a selected DMD. It also sends a release request to an old current VDMME. At the same time, the selected DMD sends an MN context request to an old current DMD and becomes a new current DMD after receiving all information about the MN. Then, a current VDMME updates the MN context and sends it to a new current DMD.

3.2 Algorithm based on the response threshold model

To accomplish autonomous decision making, we adopt a biologically-inspired mathematical model of division of labours in insect societies [2]. In the model, the probability with which an idle individual i becomes a worker is given by $s^2/(s^2+\theta_i^2)$, while a worker stops performing a task with probability q. The scalar s is called stimulus, corresponding to the demand that a task is performed. s increases with the certain constant rate while decreases proportionally to the number of workers. θ_i is a threshold of individual i, meaning the degree of hesitation in performing a task.

We regard VDMMEs and DMDs as individuals and mobility management of an MN as a task and extend the model to incorporate other factors. First, for the h-th request, MN calculates two stimulus values $S_V(h)$ and $S_D(h)$ for VDMME selection and DMD selection, respectively. A request message carries both of $S_V(h)$ and $S_D(h)$ and is forwarded to member servers. The stimulus $S_V(h)$ is derived as,

$$S_V(h) = S_V(h-1) + \delta_V - \alpha_V(h), \quad (1)$$

$$\alpha_V(h) = 0.5\{D_V(h) + L_V(h)\}, \quad (2)$$

where $D_V(h)$ and $L_V(h)$ express the goodness of the current VDMME regarding the response delay and the load status, respectively. $D_V(h)$ is given as,

$$D_V(h) = \frac{\sum_{k=1}^{\min(h,W)} \rho\{d_{Vmax}(h-k) - d(h-k)\}}{\sum_{k=1}^{\min(h,W)} \rho\{d_{Vmax}(h-k) - d_{Vmin}(h-k)\}}, \quad (3)$$

where W is a window size. $d(h)$ is observed response delay of the h-th request. $d_{Vmax}(h)$ and $d_{Vmin}(h)$ are the maximum and minimum estimated response delay among all member VDMMEs, respectively.

$L_V(h)$ is defined by Jain's fairness index as,

$$L_V(h) = \frac{\{\sum_{j=1}^{n_V} U_j(h-1)\}^2}{n\sum_{j=1}^{n_V} U_j(h-1)^2}, \quad (4)$$

where n_V is the total number of VDMMEs in a network and $U_j(h-1)$ is the utilization of VDMME j as a result of the $(h-1)$-th allocation. The utilization of a VDMME is defined as the β-th power of the ratio of the number of serving MNs to the maximum number, i.e. capacity. APs periodically collect load information of all servers and the information is embedded in a response message it forwards to an MN.

The stimulus $S_D(h)$ of DMD is derived in the similar way with the case of VDMME with a difference in $\alpha_D(h)$.

$$S_D(h) = S_D(h) + \delta_D - \alpha_D(h), \quad (5)$$

$$\alpha_D(h) = 0.5\{D_D(h) + L_D(h)\} \times \{1 + N(h)\}, \quad (6)$$

where $N(h)$ is the number of DMD migrations that an MN observed in past W requests.

For autonomous decision making, each server maintains a set of X_i, θ_i, and O_i for MN i. When a server is a candidate, $X_i = true$. Otherwise $X_i = false$. θ_i is a threshold value ($0 \leq \theta_i \leq \theta_{max}$) indicating the degree of hesitation in serving MN i. O_i expresses the operational mode. When server i is serving MN i as a current server, $O_i = active$. Otherwise

$O_i = idle$. Therefore, for MN i, only one pair of VDMME and DMD have $X_i = true$ and $O_i = active$ in a network.

When servers whose X_i is $false$ receives the h-th request from MN i, it becomes a candidate with probability below.

$$P(X_i(h) = false \rightarrow X_i(h) = true) = \frac{S_i(h)^2}{S_i(h)^2 + \theta_i(h)^2 A_i(h)}. \quad (7)$$

$S_i(h)$ is $S_V(h)$ at a VDMME and $S_D(h)$ at a DMD. $A_i(h)$ biases the threshold. A VDMME derives $A_i(h)$ as,

$$A_i(h) = w_d d_i(h) + w_u U(h), \quad (8)$$

where w_d and w_u are weighting parameters. $d_i(h)$ is normalizd delay derived by $d_i(h) = 1 - D(h)$, in which $D(h)$ is calculated similarly to $D_V(h)$ but using the estimated delay of the VDMME. $U(h)$ is the utilization of the VDMME.

On the contrary, a DMD calculates $A_i(h)$ as,

$$A_i(h) = w_f F_i(h) + w_d d_i(h) + w_u U(h), \quad (9)$$

where $F_i(h) = 1 - e^{-(b-H)^2/W}$ is related to the mobility of a requesting MN. H is the number of MN migration between cells. b is defined depending on a location of a DMD, i.e. 0 for AP, 3 for LO, and 8 for RO. When H is close to b, $F_i(h)$ is small and the probability to become a candidate is high. Thus, for example, a DMD in an AP becomes a candidate with high probability for an immobile MN.

On receiving a request, a server whose $X_i(h-1) = true$ changes to $X_i(h) = false$ at the constant quitting probability q_i ($0 \leq q_i \leq 1$) on receiving a request from MN i. It contributes to task rotation among servers. We also adopt a reinforcement mechanism of the response threshold model. More specifically, a server adjusts the threshold as,

$$\theta_i(h+1) = \begin{cases} \theta_i(h) - \xi, & \text{if} X_i(h) = true \\ \theta_i(h) + \phi, & \text{if } X_i(h) = false \end{cases} \quad (10)$$

where ξ and ϕ are parameters of the speed of differentiation.

4. Evaluation

In this section, we evaluate our proposal from viewpoints of response delay, the number of DMD migrations, and the number of candidates.

4.1 Simulation Setting

A mobile core network consists of one RO for 3 TAs. There are 7 LOs / TA and 7 cells / LO. As in many other papers, each AP covers a hexagonal cell and 7 cells organizes a hexagonal TA [6][7], in total, there are 21 LOs and 147 APs. We consider torus topology and each TA shares borders with all the other TAs. There are 3 DMDs in a RO, each of which corresponds to a TA. There are 3 DMDs and 5 VDMMEs in an LO and one VDMME and one DMD in an AP, respectively. The maximum number of MNs that each VDMME and DMD can manage are 500 and 8000 respectively. Those numbers are chosen considering different capacity of locations, but can be any arbitrary numbers. Propagation delays are set at 2 ms, 5 ms, and 15 ms between an MN and an AP, between an AP and an LO, and between an LO and a RO, respectively, taking into account their physical distances. Furthermore, delays between neighboring APs and between neighboring LOs are set at 4 ms and 10 ms, respectively.

Table 1. parameter setting

Para.	Description	Value
θ	threshold	0.5
ξ	threshold adaptation parameter	0.01
ϕ	threshold adaptation parameter	0.1
δ_V	increasing rate of demand for VDMME	0.5
δ_D	increasing rate of demand for DMD	0.6
q_i	quitting probability	0.01
W	history window	5
w	weighting values in Eq.(8) & (9)	10
β	exponent of utilization derivation	3

The number of MNs is 14700 and they are equally distributed over cells at the beginning of simulation. All MNs are always attached but only 30% of them are randomly selected to be connected at every time step, which is 10 mins long. To simulate mobility, each MN has a stay timer. When a stay timer expires, an MN moves to a random neighbor cell. A timer interval is randomly set following the Gaussian distribution with average of T_s and variance of 1. An initial timer value is randomized to avoid synchronization. MNs are divided into four mobility levels of equal size: immobile, low mobility (T_s=10 hrs), medium mobility (2 hrs), and high mobility (0.5 hrs). A TAU timer is set at 30 mins and initial values are randomized. Initially, all MNs are allocated servers at the nearest AP as current servers.

In the next section, we show results of response delay, i.e. delay from emission of a request to reception of a response, the number of DMD migrations per request, and the number of candidates as an indicator of C-plane overhead, measured at the end of a simulation run of 80 simulation hrs averaged over 100 runs. The latter two are used as the rightmost value in figures is an average of all MNs with different mobility. Parameters used in our proposal are summarized in Table 1. As a comparative method, we imitate a DMM scheme [4] by persistently allocating DMDs in a RO to MNs. We call this method DMM-RO.

4.2 Results and Discussion

In Fig. 2, we can easily find that our proposal achieves much lower delay than DMM-RO. The average delay for immobile MNs is about 4 ms, which implies that our proposal successfully allocates servers in a nearest AP to an MN. Regarding MNs with low mobility, the average delay is as large as 5.79 ms. It is because some MNs are allocated a DMD in an LO as we expected in Section 2. Similarly, appropriate allocation is performed for MNs with medium or high mobility. Although details are not discussed here for space limitation,, an MN with higher mobility is likely to be allocated a DMM in an LO or a RO to avoid frequent DMM migration sacrific-

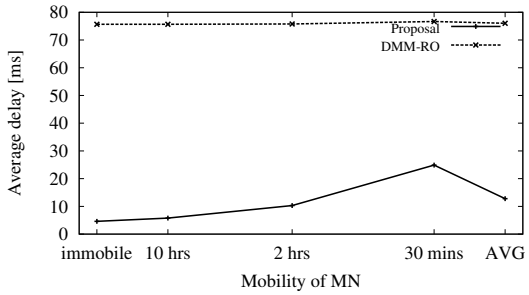

Figure 2. Average response delay

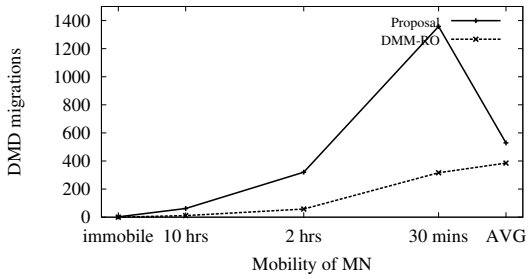

Figure 3. Number of DMD migrations

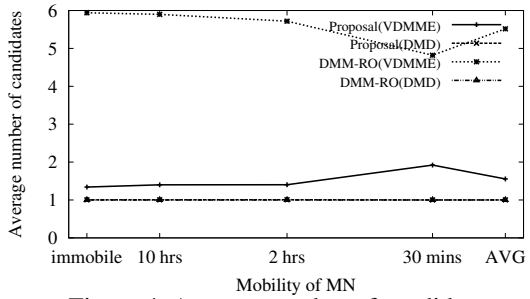

Figure 4. Average number of candidates

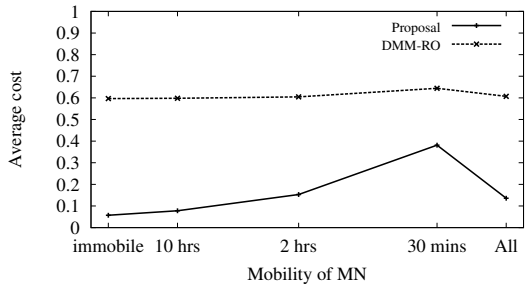

Figure 5. Average cost

ing delay. Such near optimal allocation is achieved without sacrificing fairness. Our proposal accomplished about 0.64 for VDMME and 0.72 for DMD in Jain's fairness index, while it is 0.59 and 0.014 in DMM-RO.

Figure 3 shows that the number of DMD migrations increases as the mobility becomes higher. In addition, our proposal results in more frequent DMD migrations than DMM-RO. Because of the fixed quitting probability, candidates, do not quit even if they are not appropriate any more. As a result, the number of candidates becomes large and selection is likely to be random. As future work, we should dynamically adjust the quitting probability to solve the problem.

The number of candidate VDMMEs of our proposal is smaller than that of DMM-RO as shown in Fig. 4. Since candidates send notification messages, the redundancy results in waste of bandwidth. In DMM-RO, the response delay cannot be reduced due to distant DMD and $\alpha_i(h)$ is always smaller than δ_i. Consequently, stimulus S_V keeps increasing and many member VDMMEs become candidates.

Finally in Fig. 5 we show comparison by cost, which we empirically define as the sum of normalized values of three measures. The normalized average delay N_D is given as (average delay - minimum delay) / (maximum delay - minimum delay), where maximum delay is 84 ms and minimum delay is 4 ms. The normalized number of DMD migrations N_M is given as the ratio of the number of DMD migrations to the number of requests. The normalized number of candidates N_C is the average of ratios of the number of candidates to the number of members regarding VDMME and DMD. Then, cost is derived as $N_D/2 + N_M/4 + N_C/4$, considering the importance of delay minimization. As shown, cost of our proposal is much lower than DMM-RO.

5. Conclusion

In this paper, we propose a novel flat mobile core network architecture together with an autonomous and adaptive server selection scheme. Simulation results show that our proposal can achieve near optimal delay, but the number of DMD migrations is not small enough. We plan to introduce an algorithm of quitting probability adaptation to limit candidates to the minimum and necessary number. We also need to compare our proposal with other schemes under more complex and realistic mobility scenario.

References

[1] 3GPP; service requirements for machine-type communications (MTC); stage 1. TS 22.368 V12.3.0, 3GPP, Dec. 2013.

[2] E. Bonabeau, A. Sobkowski, G. Theraulaz, and J. L. Deneubourg. Adaptive task allocation inspired by a model of division of labor in social insects. In *Proceedings of Biocomputing and Emergent Computation*, pages 36–45, 1997.

[3] H. Chan, D. Liu, P. Seite, H. Yokota, and J. Korhonen. RFC 7333: Requirements for distributed mobility management. Aug. 2014.

[4] H. Chan, H. Yokota, J. Xie, P. Seite, and D. Liu. Distributed and dynamic mobility management in mobile internet: Current approaches and issues. *Journal of Communications*, 6(1):4–15, Feb. 2011.

[5] F. Giust, A. D. la Oliva, and C. Bernardos. Mobility management in next generation mobile networks. In *Proceedings of IEEE WoWMoM*, pages 1–3, June 2013.

[6] R. Langar, N. Bouabdallah, and R. Boutaba. A comprehensive analysis of mobility management in MPLS-based wireless access networks. *IEEE/ACM Transactions on Networking*, 16(4):918–931, Aug. 2008.

[7] J. Li, H. Kameda, and K. Li. Optimal dynamic mobility management for PCS networks. *IEEE/ACM Transactions on Networking*, 8(3):319–327, June 2000.