A control method for autonomous mobility management systems toward 5G mobile networks

Daichi Kominami*, Takanori Iwai†, Hideyuki Shimonishi†, and Masayuki Murata‡
*Graduate School of Economics, Osaka University, Japan
Email: d-kominami@econ.osaka-u.ac.jp
†Graduate School of Information Science and Technology, Osaka University, Japan
Email: murata@ist.osaka-u.ac.jp
‡System Platform Research Labs, NEC Corporation, Japan
Email: t-iwai@hx.jp.nec.com, h-shimonishi@cd.jp.nec.com

***Abstract*—Fifth-generation mobile and wireless communication systems are actively being studied as a next-generation mobile communication network. People and devices will connect to the Internet via wireless networks in future communication networks, but many challenges must be solved to realize 5G networks. For example, it is difficult to manage connected user devices using centralized control in the control plane. We previously proposed an architecture for autonomous and distributed mobility management and a biology-inspired mobility management scheme that adaptively selects the location of management entities. However, this scheme has some problems from a network-wide perspective, due to entities' autonomous decision-making. This paper introduces a control node for monitoring and managing the network to resolve the otherwise unsolvable problem of autonomous distributed control. We show that this control node can improve network stability without much loss of performance in the entire network.**

Keywords-5G network, attractor selection, bio-inspired algorithm, controlled self-organization

I. Introduction

Future communication networks will connect people and devices via wireless networks in the so-called Internet of Things (IoT), and machine-to-machine (M2M) communication will be critical for secure, safe, and affluent living. To that end, fifth-generation (5G) mobile and wireless communication systems have attracted attention as a next-generation mobile communication network.

Many technical challenges remain for 5G networks [1], [2]. To enhance network resource utilization and network performance in the IoT and M2M communication, 5G networks should allow for the transmission of many small packets with low latency and small communication overhead. In current long-term evolution/evolved packet core (LTE/EPC) networks, a serving gateway (SGW) handles the user plane (U-Plane) and a mobility management entity (MME) manages the control plane (C-Plane) in a centralized manner. Such centralized information management of connected user equipment (UE) has caused bottleneck problems [3].

The control-plane load will grow in the future due to the periodic background traffic generated by many mobile devices and various types of traffic generated by a lot of IoT/M2M devices. The concentration of the load on the control plane results in the increase in delay in the control plane, which causes the performance degradation of the mobile network as a whole. Fully autonomous and distributed control techniques, called self-organization control techniques, can resolve such problems. Self-organized control is described as a time-evolution equation based on local information for deriving a solution, which avoids rapid increases in the amount of collected information and calculated solutions. Self-organized control optimizes the system through the emergence of the individual decision-making.

We have previously proposed a network architecture that employs MMEs as a logical function in the mobile network [4]. These MMEs autonomously perform UE management in a distributed manner as autonomous distributed MMEs (ADMME). The role of mobility management for a UE can be delegated from one ADMME to another (called *ADMME switching*), and the selection of a new ADMME is made at the previous ADMME's discretion. We call this *ADMME selection*. Determining the best placement of ADMMEs in a mobile-core network and determining which ADMME manages which UE by considering the UEs' position are quite challenging problems.

ADMME was proposed to distribute MME function and achieve delay reduction and load balancing in the control plane. Similar efforts are being made in the *distributed MME* [5]. However, in [5], it is necessary to preallocate servers for control-plane usage, and dynamic load balancing among servers is not considered. ADMME performs the dynamic migration of UE, and manages UE in an autonomous and decentralized manner taking the delay and load of the surrounding MME into consideration.

In [4], an ADMME selection method based on the *attractor-selection algorithm* [6] was proposed for load balancing between ADMMEs and for suppressing communication delay between an ADMME and its UEs, where each ADMME makes decisions using only local information. However, autonomous and distributed control generally

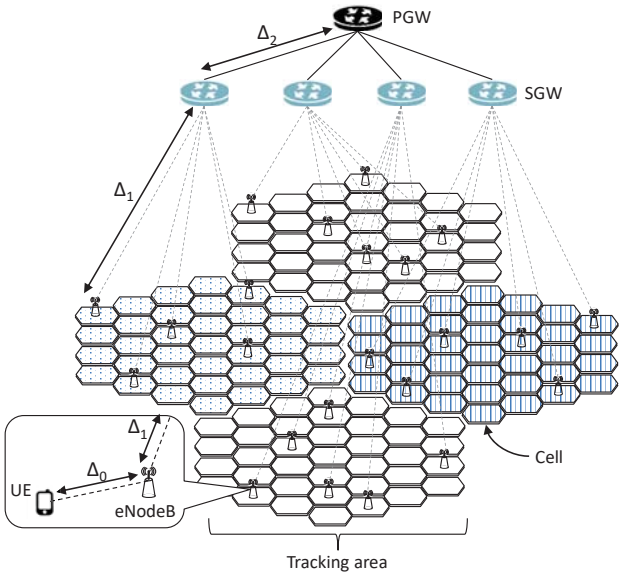


Figure 1. Network and delay model

suffers from instability when there exist multiple states that satisfy the local optimality [7]. Our proposed method in [4] cannot solve this problem and furthermore generates excessive ADMME switching. Movement of the context and an increase in signaling accompanying this switching increased load in the C-Plane. In addition, the local decision-making of ADMMEs cannot directly control the performance of the entire network.

The main goal of this paper is to propose a novel network architecture that solves problems in the autonomous ADMME selection method while retaining its advantages. Note that our proposal is designed to suppress ADMME switching even when considering UE mobility. We introduce a *control node* [8] that monitors and manages the network to meet performance requirements and suppress instability. The control node described in this paper induces network performance to a sub-optimal value and suppresses connection switching between an ADMME and a UE, so long as doing so does not result in deterioration from the desired performance. To that end, the control node has a feedback mechanism, where it periodically observes the network performance and provides control input to each ADMME according to the observed performance.

The remainder of this paper is organized as follows. We first briefly explain attractor selection-based on the ADMME selection algorithm in Section II. We then propose a control architecture for an autonomous mobility management system in Section III. A computer simulation described in Section IV demonstrates the performance of the proposed method. Finally, Section V concludes this paper and suggests areas for future work.

II. Autonomous ADMME Selection Based on Attractor Selection

An attractor selection-based ADMME selection algorithm is proposed in [4]. We first briefly explain the ADMME selection problem, and then explain how the attractor selection algorithm solves this problem.

A. ADMME Selection Problem

An ADMME can run on any eNodeB, SGW, or PGW node. As in [4], this paper assumes that all nodes have ADMME functionality. Each ADMME manages the mobility information of UEs. ADMME switching can happen when a UE transmits a tracking area update (TAU) request, a handover request, or an attach request to a current ADMME. Here, a TAU request is transmitted when the TAU timer of a UE expires or a UE moves to another tracking area. A handover request starts when a UE moves to another cell. An attach request is sent when a UE joins the mobile network.

When an ADMME receives such a request from a UE, the ADMME selects an ADMME that is eligible for transferring management of the requesting UE to reduce communication delays between the UE and the new ADMME and to reduce the load concentration on the new ADMME. The ADMME selects a new ADMME from a candidate set that consists of ADMMEs in all nodes on the path between a requesting UE and the current ADMME. The candidate set also includes ADMMEs in the SGW and the PGW nearest to the UE. In [4], the ADMME selects a new ADMME from the candidate set according to the *delay history* and *load status* of nodes (details are described in the following subsection). We assume that communication delay is estimated by a node that has an ADMME through the request of a UE. Each ADMME periodically collects the load status of nodes in the current candidate set. After determination of a new ADMME, the current ADMME transmits a delegation message to the new ADMME with context information of the requesting UE. The new ADMME responds to the UE's request with a message.

B. Attractor Selection-based ADMME Selection

The attractor selection model mathematically explains how biological systems adapt themselves to unexpected changes in their surroundings [6]. In this model, each system component periodically updates its state. This model therefore provides not a simple heuristic algorithm, but remarkable adaptation to dynamically changing environments according to the fitness to the current environment. Various network control methods have applied this [9]. In [4], we applied this algorithm to solve the ADMME selection problem for delay reduction and load balancing. Figure 1 shows our assumed network model. For simplicity, we assume that delays between a UE and an eNodeB (Δ_0), between an eNodeB and an SGW (Δ_1), and between an SGW and a PGW (Δ_2) are a constant value. The load status is the number of UEs that the corresponding ADMME manages.

An ADMME has a vector $\mathbf{m} = (m_1, m_2, \cdots, m_M)$ for each UE that the ADMME manages. M is the cardinality

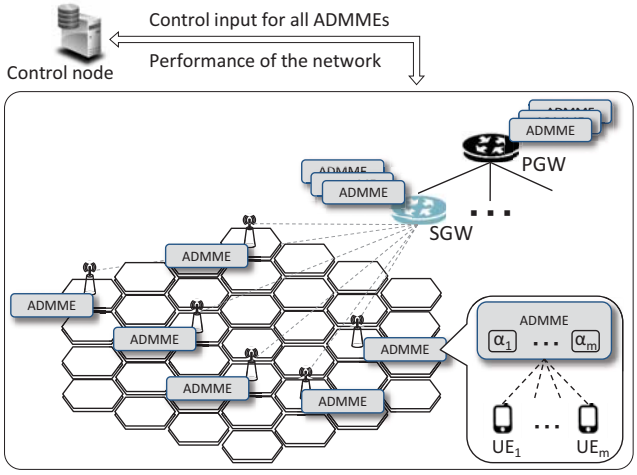


Figure 2. Control node and ADMMEs in the assumed network

of the candidate set for the corresponding UE. m_i is a state value that corresponds to the adequacy of ADMME i for selection. An ADMME also has a scalar value α for each UE, which is called its *activity*. Activity expresses the fitness of the current selection of an ADMME. In the attractor selection algorithm, $\mathbf{m}$ is updated as

$$\frac{\mathrm{d}\mathbf{m}}{\mathrm{d}t} = \alpha f(\mathbf{m}) + \boldsymbol{\eta}, \tag{1}$$

where f is the derivative of a function that has M attractors and $\boldsymbol{\eta}$ is noise.

When activity is high, $\mathbf{m}$ converges to an attracted state, and it randomly seeks another state when activity is low. An ADMME updates its activity when a request from a UE arrives, and it also updates $\mathbf{m}$ using the activity. The ADMME in the candidate set with the largest state value is then selected as the new ADMME. The state vector, activity, and delay history are transmitted to the ADMME that is newly taking control.

Upon the h-th request, the ADMME calculates its activity α as

$$\alpha(h) = \rho \cdot \alpha_d(h) + (1-\rho) \cdot \alpha_l(h), \tag{2}$$

where ρ takes a value in $[1 \ldots 0]$ that determines the weight of activities $\alpha_d(h)$ and $\alpha_l(h)$, which are described below.

In attractor selection-based ADMME selection, ADMMEs use an estimated delay $\hat{d}$. ADMMEs estimate $\hat{d}_i$ as the expected communication delay from node i to a UE via the ADMME. If node i is on the shortest path between the UE and the ADMME, $\hat{d}$ is the combined delay from the UE to the ADMME and then to node i. Otherwise, $\hat{d}_i = 0$. Delay information for the previous W steps is stored as a delay history. To maintain this, while collecting the load status of nodes in the current candidate sets, ADMMEs periodically send probe packets to nodes in their candidate set. Note that in the candidate set, a larger $\hat{d}_i$ indicates closer proximity of node i to the corresponding UE.

The activity based on $\hat{d}$, $\alpha_d(h)$, is calculated as

$$\alpha_d(h) = \left(\frac{\sum_{k=1}^{W} \frac{\hat{d}_{cm}(h-k)}{k}}{\max_{1 \le i \le M} \sum_{k=1}^{W} \frac{\hat{d}_i(h-k)}{k}} \right)^{\epsilon}, \tag{3}$$

where $\hat{d}_{cm}$ is the delay of the current ADMME and ϵ is a parameter to determine the output level for $\alpha_d(h)$. Activity based on load status $\alpha_l(h)$ is calculated as

$$\alpha_l(h) = \frac{\min_{1 \le i \le M} l_i(h)}{l_{cm}(h)}, \tag{4}$$

where $l_i(h)$ is the latest load status of node i for the h-th request, and l_{cm} is the load of a current ADMME.

Using the $\alpha(h)$ calculated from the above $\alpha_d(h)$ and $\alpha_l(h)$, the ADMME updates $\mathbf{m}$ as

$$\frac{\mathrm{d}m_i}{\mathrm{d}t} = \frac{s(\alpha(h))}{1 + m_{max}^2 - m_i^2} - \alpha(h) \cdot m_i + \eta_i, \tag{5}$$

where $m_{max} = \max_{1 \le j \le M}\{m_j\}$, $s(\alpha) = \alpha(h)[\beta \cdot \alpha(h)^{\gamma} + 1/\sqrt{2}]$, and η_i is white Gaussian noise with mean 0 and variance 1.

III. Control Method for Autonomous ADMME Selection

It is difficult to predict performance emerging across the entire network using ADMME selection based on attractor selection, particularly due to the nonlinearity of its dynamics. Also, the algorithm cannot deal with instabilities near local optima. Connection switching between UEs and ADMMEs increases C-plane load, and the handover procedure causes delays. To solve these problems, we propose a novel architecture for controlling such systems by defining a new activity function that allows system managers to control system stability and performance.

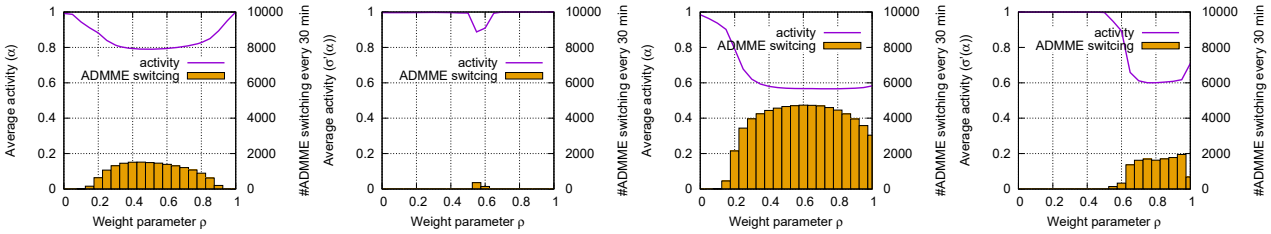
A. New activity function

The activity α in the attractor selection algorithm expresses the fitness of the current ADMME state, so α determines ADMME behavior. When α for a UE is larger than about 0.8, an ADMME does not relinquish control of the UE. When α is smaller than 0.8, an ADMME is more likely to delegate mobility management of the UE to another ADMME. Since system performance predictions are difficult, the activity α in (2) does not necessarily exceed 0.8. We therefore use a sigmoid function $\sigma(\alpha)$, defined as

$$\sigma(\alpha) = \frac{1}{1 + e^{-g(\alpha - \alpha_{th})}}, \tag{6}$$

instead of α to guarantee that activity exceeds 0.8. Here, g is a non-negative parameter that determines the decay characteristics of the sigmoid function. The sigmoid function decays more quickly with larger g. α_{th} is the inflection point of the function.

Although the sigmoid function yields the advantage of stability, it introduces another problem. When α_i, the activity for UE i, is smaller than α_{th}, the ADMME may hand a



(a) Static & autonomous (b) Static & controlled (c) Random walk & autonomous (d) Random walk & controlled

Figure 3. Activity and ADMME switching vs. weight parameter ρ

Table I
PARAMETERS FOR THE ATTRACTOR SELECTION ALGORITHM

Parameter	Value
β	10
γ	10
ϵ	2
W	5

UE i over to another ADMME even if the current ADMME is a good fit. This occurs when $\sigma(\alpha_i)$ is nearly zero, which causes less convergence of the system. To solve this problem, $\sigma(\alpha)$ should not be too small, specifically $0 \leq \alpha_i \leq \alpha_{th}$. We therefore use

$$\sigma'(\alpha) = \max\big(\alpha, \sigma(\alpha)\big) \tag{7}$$

as a new activity function and control its threshold parameter α_{th} to realize high stability and convergence speed in the system.

B. System control

As described in the previous section, an ADMME uses local information to select new ADMMEs to manage its UEs. The attractor selection model dynamically updates the system state to adapt to changing environments. When the activities of all ADMMEs are close to 1, load balancing and delay minimization are realized in all ADMMEs. However, load balancing and delay minimization among all the nodes are not compatible. If multiple objectives are incompatible, ADMME activities are unlikely to become 1 in the attractor selection-based ADMME selection algorithm. Since ADMMEs operate in an autonomous, distributed manner, they cannot know performance values of the entire network before the system operation. The attractor structure of each ADMME is determined by its activity and the performance it locally experiences.

The difficulty of predicting the system performance indicates that we cannot determine the best value of α_{th} beforehand. When α_{th} is comparatively high, frequent ADMME switching may occur if $\sigma'(\alpha)$ cannot be larger than 0.8. When α_{th} is smaller, an ADMME may stop with a relatively poor choice, which worsens system performance.

To determine the best α_{th}, a control node manages α_{th} to achieve high system stability and performance (Fig. 2). The control node should control individual ADMME behavior and network performance. In the proposed method, the control node monitors the performance of the whole network and introduces control input to each ADMME so that it satisfies performance requirements and reduces connection switching between UEs and ADMMEs.

IV. SIMULATION EXPERIMENTS

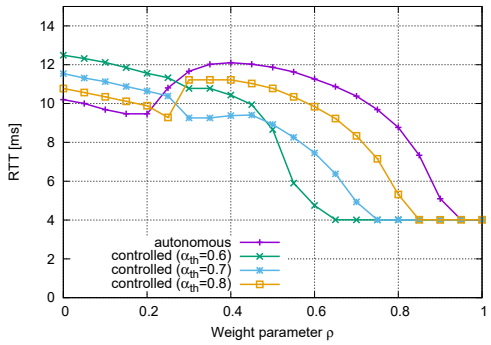
This section evaluates the proposed method using computer simulations. We show the simulation results for a purely autonomous distributed system and a controlled system. Comparing these results shows both the advantages and the disadvantages of the proposed method.

A. Settings

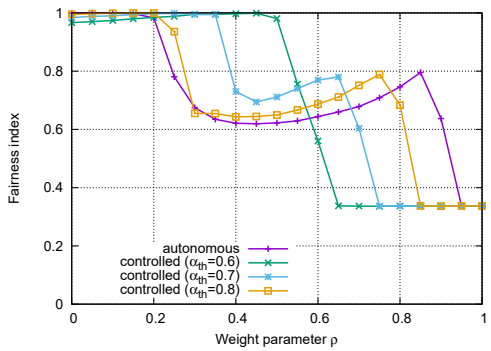
We assume a mobile core network consisting of one PGW and four SGWs. Each SGW corresponds to one TA and each TA comprises 37 hexagonal cells, as shown in Fig. 1. There is one eNodeB in each cell, so there are 148 eNodeBs in total. For simplicity, delays between nodes are static: $\Delta_0 = 2$ ms, $\Delta_1 = 20$ ms, and $\Delta_2 = 3$ ms. The number of ADMMEs in an eNodeB, an SGW, and a PGW is set to 1, 5, and 5, respectively. At the beginning of the simulation, 100 UEs are deployed in each cell and they connect to an ADMME on the nearest SGW, so each ADMME on a SGW has 740 UEs. In our simulation, the TAU timer of each UE is set to 30 min.

We consider a static pattern and a random-walk pattern. In the static pattern, no UEs move to other cells. In the random-walk pattern, each UE stays in a cell for T_s min, then moves to a neighboring cell. After moving, the UE sends a handover request to its current ADMME via an eNodeB in a new cell. T_s is set to 100 min. Table I lists parameters in the attractor selection algorithm, which are used in [4]. We set g to 30 and all ADMMEs in this evaluation use the same value of α_{th}.

Here, we describe metrics for our evaluation. As we discussed in Section II, we examine the delay, the load balancing, and the ADMME switching in the C-plane. For



(a) Average RTT between ADMMEs and UEs



(b) Fairness index

Figure 4. Performance in the immobility scenario

this, in our simulation, we show the average round-trip delay time (denoted as RTT) between ADMMEs and UEs, the Jain's fairness index among the loads of nodes, and the number of ADMME switching. Here, fairness index (FI) is calculated as follows.

$$FI(h) = \frac{(\bar{l}_{eNodeB}(h) + \bar{l}_{SGW}(h) + \bar{l}_{PGW}(h))^2}{3 \cdot \left((\bar{l}_{eNodeB}(h))^2 + (\bar{l}_{SGW}(h))^2 + (\bar{l}_{PGW}(h))^2\right)},$$

where $\bar{l}_n(h)$ is an average load status of eNodeBs, SGWs, or a PGW at current time step h ($n \in \{\mathrm{eNodeB},\ \mathrm{SGW},\ \mathrm{PGW}\}$).

B. Results

1) ADMME switching: We first show that ADMME switching occurs more frequently when α gets smaller. Here, we set α_{th} to 0.6. Figure 3 shows the average activity over ADMMEs and the average number of ADMME switches every 30 min while changing ρ. In that figure, the notation "autonomous" indicates the original ADMME selection method in [4], and "controlled" indicates the proposed method. When UEs do not move, the average α in Fig. 3(a) does not reach 1 except when $\rho = 0$ or 1. Therefore, ADMME switching occurs even in the case of immobile UEs, especially when ρ is about 0.5. This is because α_l and α_d do not simultaneously become 1 for almost all ADMMEs when ρ is not 0 or 1. By setting α_{th} to 0.6, the average value of $\sigma'(\alpha)$ exceeds 0.9 for all values of ρ, which drastically reduces ADMME switching (Fig. 3(b)). Here, the average $\sigma'(\alpha)$ is comparatively small when ρ is 0.55 or 0.60 for the same reason as in Fig. 3(a).

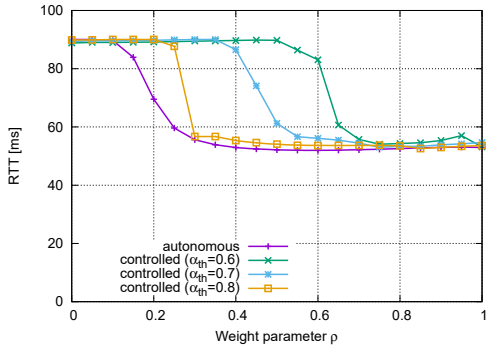
As Fig. 3(c) shows, the number of ADMME switches becomes very large as the node moves. The activities of almost all nodes become 1 only when $\rho = 1$, because load balancing does not depend on current UE positions. If ρ has any other value, the delay time between a UE and an ADMME increases as the UE moves to another cell, resulting in decreased activity. When ρ exceeds 0.3, the activity value changes little. By using a control threshold α_{th}, we can bring the average activity to almost 1 when $\rho \leq 0.5$ (Fig. 3(d)), greatly reducing the number of ADMME switches. When $\rho \geq 0.5$ the average activity is slightly larger than that in Fig. 3(c), but the number of ADMME switches can be reduced by about half, because the number of ADMMEs whose activities take values close to 1 increases. We can thus reduce the number of ADMME switches by setting an appropriate value for α_{th}. In the following section, we examine the effect of α_{th} on network performance.

2) Network performance: Figure 4 shows average RTT and fairness indices resulting from changing ρ. As this figure shows, performance in the original method irregularly varies depending on the value of ρ. The important point here is that ρ, which is a weight parameter between delay reduction and load balancing, cannot fulfill its role. This demonstrates the difficulty of predicting the performance of self-organizing systems.

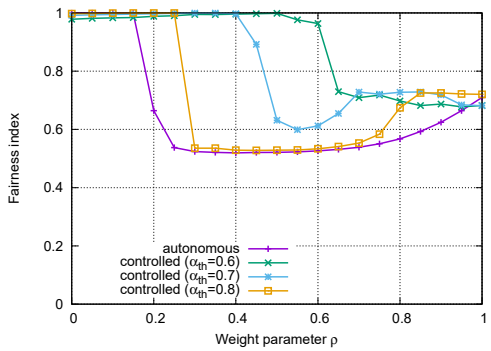
Using a control node, network managers can adjust RTT and the fairness index. The lower the value of α_{th}, the greater the influence of ρ, which makes it possible to assign different performance weights. The figure shows the results of setting α_{th} to 0.6, 0.7, and 0.8. Increased α_{th} indicates increased "autonomy." Also, if α_{th} is small, the system will tend to fall into local optima when searching for the best state, so we need to set an appropriate α_{th}. Fortunately, network managers can control α_{th} through the control node in our proposal. Network managers can confirm performance information and the number of ADMME switches, providing feedback for α_{th}.

With respect to the network performance, when activity is higher than about 0.8, **m** converges and ADMME switching is suppressed. However, as mentioned above, this leads to the system falling into local optima, so network performance may become worse than before.

Figure 4 shows the network performance without consideration of node movement. When $\rho \leq 0.2$, RTT is increased by adding control (Fig. 4(a)). In contrast, RTT decreases when $\rho \geq 0.3$. In both cases, the effect of the weight parameter ρ powerfully emerges from the control method. In the figure, as α_{th} gets smaller RTT approaches about



(a) Average RTT between ADMMEs and UEs



(b) Fairness index

Figure 5. Performance in the random-walk scenario

4 ms with smaller ρ. Here, a 4 ms RTT means that all UEs belong to ADMMEs in eNodeBs. By the control method, when $\rho \geq 0.8$ the fairness index of the control method is lower than that of the original method (Fig. 4(b)). However, setting α_{th} to a lower value brings the fairness index to about 1 within a broader range of ρ. From these results, although the original method fails to adjust the balance between delay reduction and load distribution by ρ, the proposed method improves this point, particularly when $\alpha_{th} = 0.6$.

Figure 5 describes network performance in consideration of node movement. The difference between the RTT of "controlled" and "autonomous" systems widens as compared with the case where node movement is not considered. When the node moves with a sojourn time of 100 min, the "autonomous" results show the shortest value of those cases shown in Fig. 5(a). Here, the RTT of all results approach about 52 ms, meaning that most UEs connect to SGWs' or PGW's ADMMEs. Contrary to the results for RTT, the fairness index of the control method outperforms original values for almost every ρ.

It is unknown what results will be obtained in advance even when using α_{th}, but it is important to introduce a control method that can manage them. Since the appropriate threshold value changes according to the moving speed of UEs, it is important to deal with environments in which various moving patterns coexist. One solution for this is to set an appropriate value of α_{th} for individual ADMMEs according to the frequency of node movement, which we will address in future work.

V. Conclusion and Future Work

We proposed a method for controlling network stability and performance in an autonomous system for fifth-generation mobile and wireless communication systems. We introduced a control node into the self-organized ADMME selection algorithm. Through computer simulations, we first showed that although the proposed method is very simple, the introduction of a control method positively impacts system stabilization and performance control, reducing overhead of the control plane in a 5G network. We also showed that the original attractor selection-based ADMME selection method cannot predict its performance due to its nonlinear dynamics. This suggests that control methods are indispensable for actual use of such autonomous and distributed methods. Various distributed systems can utilize the proposed algorithm. Our future work will investigate applications to mobile edge computing architectures, where virtual computing functions are dispersed over the network and users select an appropriate function for processing requests. Our mechanism can be helpful in such systems.

References

[1] A. Osseiran and F. Boccardi, "Scenarios for 5G mobile and wireless communications: the vision of the METIS project," *IEEE Communication Magazine*, vol. 52, no. 5, pp. 26–35, Feb. 2014.

[2] "Updated scenarios, requirements and KPIs for 5G mobile and wireless system with recommendations for future investigations," May 2015. [Online]. Available: https://www.metis2020.com/wp-content/uploads/deliverables/METIS_D1.5_v1.pdf

[3] RFC 7333, "Requirements for distributed mobility management," Oct. 2014.

[4] H. Yang, N. Wakamiya, M. Murata, T. Iwai, and S. Yamano, "An autonomous and distributed mobility management scheme in mobile core networks," in *Proceedings of 9th EAI International Conference on Bio-inspired Information and Communications Technologies (BICT)*, Dec. 2015.

[5] X. An, F. Pianese, I. Widjaja, and U. Günay Acer, "DMME: A distributed LTE mobility management entity," *Bell Labs Technical Journal*, vol. 17, no. 2, pp. 97–120, Aug. 2012.

[6] A. Kashiwagi and I. Urabe, "Adaptive response of a gene network to environmenal changes by fitness-induced attractor selection," *PLoS One*, vol. 1, no. 1, pp. e49:1–10, Dec. 2006.

[7] Q. Wu, G. Ding, Y. Xu, S. Feng, Z. Du, J. Wang, and K. Long, "Cognitive internet of things: a new paradigm beyond connection," *IEEE Internet of Things Journal*, vol. 1, no. 2, pp. 129–143, 2014.

[8] D. Clément and B. Cyrille, "Toward organic computing approach for cybernetic responsive environment," *arXiv*, vol. 1601.01614, Jan. 2016.

[9] K. Leibnitz and M. Murata, "Attractor selection and perturbation for robust networks in fluctuating environments," *IEEE Network*, vol. 24, no. 3, pp. 14–18, May 2010.