Dynamic Probabilistic Caching Algorithm with Content Priorities for Content-Centric Networks

Warit Sirichotedumrong, Wuttipong Kumwilaisak, Saran Tarnoi, and Nattanun Thatphitthukkul

This paper presents a caching algorithm that offers better reconstructed data quality to the requesters than a probabilistic caching scheme while maintaining comparable network performance. It decides whether an incoming data packet must be cached based on the dynamic caching probability, which is adjusted according to the priorities of content carried by the data packet, the uncertainty of content popularities, and the records of cache events in the router. The adaptation of caching probability depends on the priorities of content, the multiplication factor adaptation, and the addition factor adaptation. The multiplication factor adaptation is computed from an instantaneous cache-hit ratio, whereas the addition factor adaptation relies on a multiplication factor, popularities of requested contents, a cache-hit ratio, and a cache-miss ratio. We evaluate the performance of the caching algorithm by comparing it with previous caching schemes in network simulation. The simulation results indicate that our proposed caching algorithm surpasses previous schemes in terms of data quality and is comparable in terms of network performance.

Keywords: Addition factor adaptation, Caching algorithm, Content-centric network, Content popularity, Data priority, Multiplication factor adaptation.

I. Introduction

Content-Centric Networking (CCN) has been proposed in [1]–[3] as a promising Future Internet Architecture (FIA). CCN offers a new paradigm that addresses data by content rather than location. CCN identifies actual content via the content name ("name") and easily retrieves the content from sources that may not be the original storage location. Within CCN, the router ("CCN router") has its own memory to cache the content. The content requester sends an "interest packet" to the CCN router to retrieve the "data packet," so the corresponding data packet can be retrieved from either the original servers or the CCN routers. A CCN router contains three main components: 1) a Forwarding Information Base (FIB); 2) a Pending Interest Table (PIT); and 3) a Content Store (CS). In general, by caching data packets in the CS, the content is moved closer to the requester than the current Internet architecture. In reality, the CS memory is limited, and so the caching algorithm must decide whether the content must be cached.

Various caching schemes for CCN have been proposed in [1]–[21]. A caching scheme that caches every incoming data packet was originally proposed as a universal caching scheme [1]–[3]. Caching schemes based on content popularities were presented in [4]–[8]. Caching schemes based on energy awareness were introduced in [9]–[11]. Furthermore, there are several probabilistic caching schemes [12]–[19] whose key idea is to let routers randomly cache incoming data packets. However, the previous caching schemes did not consider the unequal importance of the data packets. Prioritized probabilistic

Manuscript received Feb. 16, 2017; revised Apr. 16, 2017; accepted Apr. 24, 2017.

Warit Sirichotedumrong (sir.warit@gmail.com) and Wuttipong Kumwilaisak (corresponding author, wuttipong.kum@kmutt.ac.th) are with the Department of Electronics and Telecommunication Engineering, King Mongkut's University of Technology Thonburi, Bangkok, Thailand.

Saran Tarnoi (sarantarnoi@gmail.com) is with the Department of Information Technology, CPPC Public Co., Ltd, Bangkok, Thailand.

Nattanun Thatphithakkul (nattanun.thatphithakkul@nectec.or.th) is with the Accessible Innovation and Universal Design Laboratory, National Electronics and Computer Technology Center, Pathum Thani, Thailand.

caching for CCN was proposed in [18] to handle the heterogeneous priorities of content, but it adjusted caching probability based on the accumulated content popularity. However, the proposed caching algorithm utilizes network events to identify content popularity [19] and provides a mathematical model to calculate the factors associated with caching probability adaptation. Although caching strategies for layered video content were proposed to deal with different video layers [20]–[21], these do not include a method for adapting to the non-stationary nature of both network environments and data content.

Based on the challenges of time-varying content and network circumstances, the contributions of this work are:

- We propose a dynamic probabilistic caching algorithm with content priorities for CCN that caches incoming data packets based on their priority and popularity.
- We formulate a mathematical model for associated factors based on popularities of requested content, cache-hit ratio, and cache-miss ratio.
- We evaluate the performance of the proposed caching algorithm and compare it with the static probabilistic caching algorithm (Prob) [14]–[17] and adaptive probabilistic caching (Aprob) [19].

The rest of the paper is organized as follows. Section II introduces the system model. Section III presents the design objectives and evaluation metrics. Section IV elaborates the dynamic caching algorithm, beginning with an explanation of the caching operation concept followed by a description of caching probability adaptation. The performance evaluation and results discussion are in Section V. Concluding remarks are in Section VI.

II. System Model

The system model in this paper consists of a caching scheme and content request model, which will be described in the following sections.

1. Caching Scheme

According to CCN architecture, CCN routers cache data packets in memory to potentially satisfy future consumer requirements. The caching system in a CCN router includes two main components: 1) a caching decision algorithm; and 2) a cache replacement policy. A caching decision algorithm decides which incoming data packets will be cached; whereas a cache replacement policy decides which cached data packets will be discarded when there is not enough room for caching new incoming data packets.

We propose the Dynamic Probabilistic Caching Algorithm with Content Priorities (D_P) as a new caching management algorithm. The proposed algorithm comprises three main components: 1) a caching operation; 2) a cache replacement policy; and 3) a notification packet. A CCN router randomly caches an incoming data packet based on a dynamic caching probability that is adjusted according to data packet priorities and the previous cache event. A notification packet notifies CCN routers when the content popularity exceeds an acceptable threshold. The algorithm utilizes Least-Recently Used (LRU) as the replacement policy.

2. Content Request Model

In real networks, there are many consumers demanding various kinds of contents. To satisfy the perpetual needs of consumers, random caching of content is utilized in the proposed caching algorithm. In this paper, we assume that there are M different pieces of content, which are requested randomly based on the Zipf-Mandelbrot distribution [18], [22]–[23]. Each piece of content is segmented into k chunks, with each chunk sliced into l layers representing l data priorities. To request the content, the consumer will send $l \times k$ interest packets to the network. For example, when consumer A demands content named "/contentA," which consists of three chunks, where each chunk is sliced into two layers, A will send six interest packets in total. We also assume that each chunk is independent of other chunks. For example, the absence of chunk m will not affect the reconstruction of chunk $m + 1$.

III. Design Objectives and Evaluation Metrics

1. Design Objective

There are many types of content that are transferred through global computer networks, such as text, audio, image, and video. The contents have unequal importance. For example, among the various types of content, audio has more priority than video and text. If a consumer is streaming a movie with subtitles, the consumer expects the best quality video, audio, and subtitles. When the network has sufficient bandwidth and there are no packet losses, the consumer can receive the highest video quality. When packet loss occurs, the consumer wants to continue to watch video, even with lower quality. In scalable video compression, video compression bit streams consist of a base layer and several enhancement layers. The base layer represents the lowest video quality that a video bit stream can provide. The video quality is improved with the addition of enhancement layers. To this end, the caching network in CCN should maintain the multimedia

https://doi.org/10.4218/etrij.17.0117.0117

communication session by preserving content reconstructed for customers with its available resources. Therefore, the proposed caching algorithm should treat content with different priorities differently.

Another objective of a caching network in CCN is to satisfy the requester with the most suitable corresponding content. As the content popularity and the requesting characteristics of consumers are generally dynamic and unpredictable, CCN routers obscurely decide which content needs to be stored in the router memory. If the demand for content is rapidly changing, content popularity will change as well. Therefore, a caching network should adapt its caching characteristics to the variations in the requesting of content.

2. Evaluation Metrics

In this paper, performance evaluation metrics both in data quality and in network performance are used to measure the efficiency of the proposed algorithm. The data quality is measured via the Normalized Information Value (V), whereas the network performance metrics are cache-hit ratio, server load, and network traffic.

A. Normalized Information Value

We define Normalized Information Value (V) as a mathematical expression to estimate the content reconstruction quality at the receiver. It is the ratio of the number of data packets received within the transmission deadline (τ) and the total number of data packets arriving at the requesters. Because content may consist of several layers, where the base layers are the most important layers and enhancement layers are for video quality improvement, the enhancement layers are useless for data reconstruction without the base layer.

To compute V at time t, define a vector of the number of received packets for all layers of chunk m that are delivered to requesters within τ seconds. This can be expressed as

$$N_m^t = [\, n_{0m}^t \quad n_{1m}^t \quad \cdots \quad n_{(l-1)m}^t \,]^T, \tag{1}$$

where n_{im}^t is the number of the ith layer data packets.

Next, we define a vector used for identifying whether or not data layer packets of chunk m are useful for data reconstruction within the transmission deadline. It can be written as

$$Q_m^t = [q_{0m}^t \quad q_{1m}^t \quad \cdots \quad q_{(l-1)m}^t]^T, \tag{2}$$

where q_{im}^t is the useful identification of the ith data layer of chunk m at time t. By default, q_{im}^t is set to 0. If the ith data layer of chunk m was delivered within the transmission deadline and the lower layers j, where $j < i$, are successfully transmitted, q_{im}^t will be equal to 1. Else, q_{im}^t is set to zero.

We define the priorities of all data layers as a weighted vector, which is represented by γ_m^t. To reflect the effects of data layers on reconstructed data quality, γ_m^t can be expressed as

$$\gamma_m^t = [\gamma_{0m}^t \quad \gamma_{1m}^t \quad \cdots \quad \gamma_{(l-1)m}^t]^T, \tag{3}$$

where γ_{im}^t is the weighted value of the *ith* data layer of chunk m at time t. The higher the weighted value, the more important the corresponding data layer is to the final data reconstruction. In general, $\gamma_{0m}^t > \gamma_{1m}^t > \ldots > \gamma_{(l-1)m}^t$.

Finally, define K_m^t as the total number of transmitted packets of chunk m at time t. This can be expressed as

$$K_m^t = [k_{0m}^t \quad k_{1m}^t \quad \cdots \quad k_{(l-1)m}^t]^T, \tag{4}$$

where k_{im}^t is the total transmitting number of ith layer data packets of chunk m at time t.

At time $t + \tau$, we can compute the useful packets of chunk m of the ith layer as

$$n_{im}^{t+\tau} = n_{im}^t \times q_{im}^t. \tag{5}$$

Based on (5), we can form the vector of the useful packet for chunk k at the transmission deadline as

$$N_m^{t+\tau} = [n_{0m}^{t+\tau} \quad n_{1m}^{t+\tau} \quad \cdots \quad n_{(l-1)m}^{t+\tau}]^T, \tag{6}$$

Based on the above equations, V of the prefix group can be computed by

$$V = \sum_{m=1}^{D} \frac{(\gamma_m^t)^T \times N_m^{t+\tau}}{(\gamma_m^t)^T \times K_m^t}, \tag{7}$$

where D is the number of content chunks for each prefix group. According to the delay for real-time communication [24], τ is set to 250 ms to ensure that the requester will receive an acceptable quality of content. V must be calculated for all prefix groups, and the average result of V for all prefix groups will represent overall reconstructed data quality.

B. Cache-Hit Ratio

Cache-hit ratio is utilized as the network performance metric. It represents how often data stored in a CCN cache matches the content required by requesters. When the requested content cannot be found in the cache, this is called a cache miss. Based on the above definition, the cache-hit ratio can be written as

$$H_t = \frac{\sum_{t_l=0}^{t} v_{t_l}^h}{\sum_{t_l=0}^{t} v_{t_l}^h + v_{t_l}^m}, \qquad (8)$$

where $v_{t_l}^h$ and $v_{t_l}^m$ are the number of cache hits and the number of cache misses between reference time zero and time t, respectively. A high cache-hit percentage implies that CCN routers in the network often satisfy requesters with cached content and few interest packets reach the original servers.

C. Server Load

Server load is defined as the total number of requests per second reaching the original servers. If the original servers receive the interest packets, this means that none of CCN routers in the network can satisfy requesters with cached data packets. High server load implies that the original server must process many requests and the caching scheme used in CCN routers is not efficient.

D. Network Traffic

Network traffic at every link in the network is also used as an indicator to measure the efficiency of the caching scheme. When there are many cache misses, more interest packets are forwarded to the server thus increasing network traffic. In contrast, if a small number of interest packets transverse the network, it implies that the caching scheme is efficient.

IV. Dynamic Probabilistic Caching Algorithm

This section describes in detail our dynamic caching algorithm. First, we explain overall concept of caching operation and the details for each step of the caching algorithm. Next, the adaptation of caching probability along with data priorities at the router is explained. The adaptation process comprises multiplication and addition factor adaptations.

1. Caching Operation

The caching operation comprises a lookup operation, a cache decision operation, and a notification operation. When a CCN router receives an interest packet, the lookup operation is started. The lookup operation is based on the method defined in [1]. However, our proposed algorithm extends the method by not only looking up content in the CS, but also by calculating and adjusting the caching probability based on the content in the CS. The lookup operation is shown in Fig. 1.

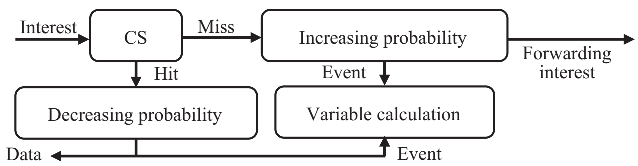

Fig. 1. Lookup operation procedure.

The lookup operation procedure can be stated as follows.

Step 1: When the CCN router receives an interest packet, it looks in its CS to find the corresponding data packet.

Step 2: If the CS contains the corresponding data packet, a cache-hit event occurs. The CCN router will transmit the corresponding data packet back to the requester. However, if requested data packet is not found in the CS, a cache-miss event occurs. The CCN router will update the new entry or an interface to its PIT.

Step 3: After cache-hit or cache-miss events are determined, they are used to compute and update the caching probability.

Figure 2 shows the cache decision operation procedure. When a CCN router receives incoming data packets, not every packet is cached in the router [1], [2]. Routers decide which packets will be cached based on the caching probability computed from the lookup operation.

The procedure can be stated step by step as follows.

Step 1: When a CCN router receives an incoming data packet, the CCN router classifies the data packet and determines its data priority.

Step 2: After classifying the data priority, the CCN router randomizes a real number from [0, 1], and compares the random number with the caching probability according to the data packet priority. If the random number is equal or less than the caching probability, the CCN router will cache the data packet; otherwise, the data packet will be discarded.

Step 3: If the CCN router decides to cache the incoming data packet, it will store the data packet to the CS and forward the data packet to every requesting consumer in the PIT.

The notification operation will start when the requested content popularity characteristics have changed. Notified packets will be broadcast to CCN routers. When the

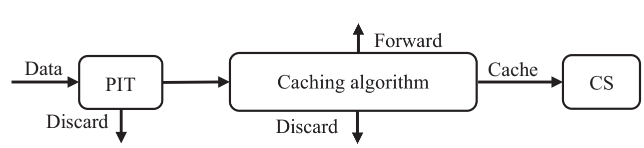

Fig. 2. Cache decision operation procedure.

routers receive the notification packets, the reinitializing process will reset the caching probability to the initial state.

2. Caching Probability Adaptation

The objective of caching probability adaptation is to adjust the caching decision along with the uncertainty of content popularities and data priorities. The caching probability is increased when the CCN router cannot satisfy an interest packet with cached content, otherwise the caching probability is decreased [19]. Instead of adjusting caching probability with static real numbers, the caching probability adaptation of the proposed caching algorithm rationally adjusts itself based on the network circumstances at each CCN router. The proposed caching probability of the ith data layer of chunk m at time $t + \tau$ can be expressed as

$$P_{im}^{t+\tau} = \begin{cases} 1 & P_{im}^{t} > 1, \\ P_{im}^{t} \times \left(f_{\mathrm{mul},im}^{t} \times \gamma_{im}^{t} \right) & \text{if hit at time } t, \\ P_{im}^{t} + \left(f_{\mathrm{add},im}^{t} \times \gamma_{im}^{t} \right) & \text{otherwise,} \end{cases} \quad (9)$$

where P_{im}^{t} is the caching probability at time t, $f_{\mathrm{mul},im}^{t}$ is a multiplication factor to adjust the caching probability, $f_{\mathrm{add},im}^{t}$is an addition factor to adjust the caching probability, and γ_{im}^{t} is the data priority at time t of the ith data layer of chunk m.

The caching probability is increased when the CS cannot satisfy the incoming interest data packet with the cached data packets. Therefore, CCN routers must cache more data packets to the CS to improve the cache-hit rate. According to the method, caching probability will be added with the multiplication of the addition factor (f_{add}) and the priority value of the content requested by the incoming interest packet. Increased caching probability means that the possibility of caching the next incoming data packet will be increased.

The caching probability will decrease when a cache-hit occurs. A cache-hit implies that the current cached data packet in the CS satisfied the demands of requesters. Therefore, it is unnecessary to increase the possibility of caching new data packets. The reduction in the caching probability can be done by multiplying the caching probability with the multiplication factor (f_{mul}) and data priority. Even if the caching probability is reduced by multiplying by f_{mul}, the CS still tries to cache the new incoming data and is ready to increase the caching probability when the network circumstances change.

However, because of the non-stationary nature of the content and the limitation of the cache size, f_{mul} and f_{add} must be adaptive to achieve better caching performance. The adaptation of these two parameters can be described as follows.

A. Multiplication Factor Adaptation

f_{mul} can be adapted via the instantaneous cache-miss ratio at time t. This factor is the most important part of the caching probability reduction process. The instantaneous cache-miss ratio is expressed in (10). f_{mul} at time t can be expressed as

$$f_{\mathrm{mul},im}^{t} = 1 - H_t, \quad (10)$$

where H_t is the instantaneous cache-hit ratio at time t.

An individual CCN router traces the number of cache hits and the number of cache-misses from the time it receives the first interest packet. The CCN router stores these numbers in memory and utilizes these two numbers when the CCN router receives the next interest packet. If there are many cache misses, it implies that the cached contents in the CCN router cannot satisfy the requests well. Therefore, the caching probability must decrease with the lower rate by multiplying with the high value of f_{mul}. When there are many cache hits, it implies that the CCN router can satisfy the requests with the cached content. Hence, the caching probability must multiply with the lower value of f_{mul}, after which the router keeps the cached content and rarely caches the incoming data packets.

B. Addition Factor Adaptation

f_{add} is a real number that is added to the caching probability to increase the possibility of caching the incoming data packet. It is based on the popularities of the requested data packets.

A popularity tracking vector is defined at time t (W^t) as a tracking vector, where each component of the vector is the number of interest packets of each prefix group. A tracking vector can be written as

$$W^t = [w_1^t \quad w_2^t \quad \dots \quad w_g^t], \quad (11)$$

where w_i^t is the number of interest packets of the *ith* prefix group at time t.

A popularity product is defined at time t (φ^t), which is the product of all elements of the tracking vector. It can be expressed as

$$\varphi^t = \prod_{i=1}^{g} w_i^t. \quad (12)$$

Note that the router calculates φ^t every t_w seconds.

f_{add} is set to 1 under three conditions. First, when it is in the initial state ($t = 0$), it means that there is no cached content in the CS, so the CS has to cache every piece of incoming content. As the probability must not exceed one, f_{add} is set to one when it exceeds one. Finally, if the requested content is totally changed, the CCN router must reset f_{add} to one. This circumstance means that the caching probability will be reset to the initial state and will cache every incoming data packet to satisfy future requests. In other words,

$$f^{t+\tau}_{\text{add},im} = 1, \text{ when } t = 0 \text{ or } f^{t}_{\text{add},im} > 1 \text{ or } \varphi^t = 0. \quad (13)$$

After the initial state, if the current state cache-miss ratio is higher than the previous state cache-miss ratio, it implies that cache-miss rate has increased; thus, the router must reduce the addition rate of the caching probability by multiplying the previous state f_{add} with the current state cache-miss ratio and waiting for future action. Our algorithm prefers to reduce the factor when the cache-miss ratio is increased because the addition factor reduction will not directly affect the caching probability. The caching probability may still increase according to the cache events.

Next, f_{add} maintains its value when the cache-miss ratio is reduced, or when the f_{add} is less than lower bound value (ε_p). The lower cache-miss ratio means that there is no need to increase or decrease f_{add}.

f_{add} is doubled when the cache-hit ratio is decreased by more than some acceptable rate of hits (ε_h) compared with the previous state cache-hit ratio. This doubling process is used to induce f_{add} to cache more content in the CS. f_{add} based on the above conditions can be expressed as

$$f^{t+\tau}_{\text{add},im} = \begin{cases} f^{t}_{\text{add},im} \times f^{t+\tau}_{\text{mul},im}, & f^{t+\tau}_{\text{mul},im} > f^{t}_{\text{mul},im} \\ f^{t}_{\text{add},im}, \quad f^{t+\tau}_{\text{mul},im} < f^{t}_{\text{mul},im} & \text{or } f^{t}_{\text{add},im} < \varepsilon_p \\ 2f^{t}_{\text{add},im}, & \frac{H_t - H_{t+\tau}}{H_t} > \varepsilon_h \end{cases}. \quad (14)$$

V. Performance Evaluation

1. Simulation Setup

This paper conducts the experiments using a computer simulator called "ndnSIM" [22], [23]. ndnSIM is a named-data networking simulator. It gathers the basic components of CCN and allows us to modify as well as to add the algorithm to the existing basic components. According to the data priorities defined in (3), the base layer, first enhancement layer, and second enhancement layer priority values are set to 1.0, 0.3, and 0.2, respectively. In this paper, we set up the experiments for Prob, Aprob, and the proposed caching algorithm. The caching probability of Prob is selected from a set of {0.1, 0.01, 0.001, 0.0001}. According to the experimental results in [19], addition and multiplication steps of Aprob are respectively set to 0.001 and 0.5. There are two types of network topology used in our caching algorithm evaluation: 1) cascading network topology; and 2) random topology.

A. Cascading Network Topology

Cascading network topology consists of a content provider, five CCN routers, and a content requester. Every node in this topology is sequentially connected as shown in the Fig. 3.

The content provider is located on the left and is connected to CCN router R1. The content requester is on the right of the topology and linked with CCN router R5. This topology accurately evaluates the performance of the caching algorithm without any disturbance from routing protocol or packet losses, so the bandwidth and propagation delay of each link are respectively assigned to 10 Gb/s and 25 ms. We evaluate our caching algorithm when the CS capacity of every CCN router is set to one percent and ten percent of data packet population. The cascading topology is evaluated for 4,000 s with two periods of content requesting. Half of the simulation time is utilized for each period. In the first period, the content requester requests 100 individual pieces of content, which are sliced into five chunks with each chunk divided into two layers. Then, the content requester changes the content requested to different groups of content, which contain 100 individual pieces of content. In the second period, the content is divided into three chunks and each chunk is classified into two layers.

B. Random Topologies

As the paper presents a new caching algorithm, we compare the proposed caching algorithm with previous caching schemes and use the new caching algorithm with three different transmission schemes. Transmission schemes used in the performance evaluation consist of Shortest Path Routing (SP) [25], Cooperative Routing

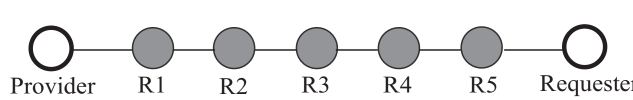

Fig. 3. Cascading network topology.

Protocol (CP) [26], and Optimal Cooperative Routing Protocol (OCRP) [27]. Shortest Path Routing finds the shortest path by which to send an interest packet, and the corresponding data packet is sent back to the requester on the same path as the interest packet. Cooperative Routing Protocol calculates the optimal path using a binary linear optimization technique to minimize the cost used by the compromising path. Optimal Cooperative Routing Protocol selectively aggregates the same prefix group interests into the same path to improve the cache utilization. OCRP calculates the optimal path by formulating binary linear optimization under three constraints: flow conservation constraint, cache contention mitigating constraint, and path length constraint.

We conduct 30 different random topologies to evaluate the caching algorithm, and each random topology contains 60 nodes. An example of random topology is shown in Fig. 4. In each topology, there are ten content providers and ten content requesters, which are randomized by the computer. A content requester requests six prefix groups. The first three groups are requested in the first period (0 s to 1,000 s), and the last three prefixes are requested in the second period (1,000 s to 2,000 s). All content in the period are divided into two chunks, and each chunk is sliced into two layers. In contrast, the leftover content in the second period is not divided into chunks, and the content is divided into three layers. In conclusion, there are 1,200 different data packets in the first period while the content requesters send the interest packets for 900 data packets in the second period of simulation. Each prefix group consists of 100 different pieces of content. The CS capacity of every CCN router is set to one percent of the first period population. To prevent and exclude packet loss, every link capacity is equal to 100 Mb/s with 25 ms for the propagation delay.

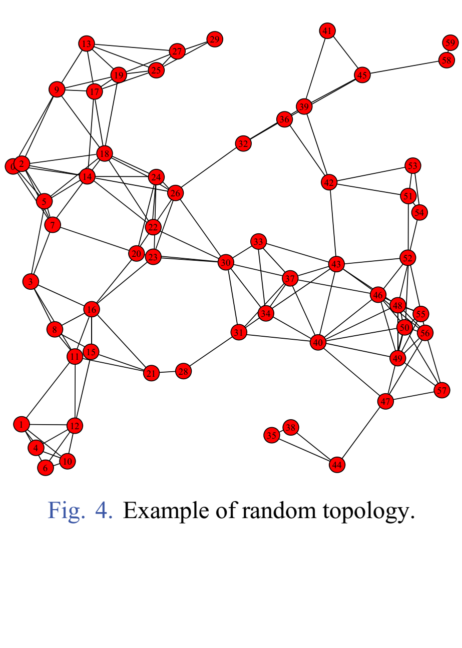

Fig. 4. Example of random topology.

2. Results and Discussions

This section presents the results of network simulation and describes the characteristics of the proposed caching algorithm compared with the previous caching schemes.

A. Cascading Network Topology

The results of this section consist of caching algorithm characteristics and simulation results. The caching algorithm characteristics describe instantaneous behavior and show how the caching algorithm is adaptively adjusted along with the cache events and designated conditions. The simulation results display the network performance evaluation expressed by V, cache-hit ratio, server load, and network traffic. To consider the effects of the CS capacity variation, we conduct the simulation differently when the CS capacity of each CCN router is equal to one percent and ten percent of content population.

a. Caching Algorithm Characteristics

To study the characteristics and the behavior of caching algorithm components, the rightmost router instantaneous components and cache-hit characteristics are first studied.

When the CS capacity is equal to one percent of content population, the characteristics of the caching algorithm components can be divided into six states. In state A, which refers to the first second of the experiment, f_{add} is equal to 1 due to the calculation process in (14), and the CCN routers always cache incoming data packets. Then, f_{add} decreases from the initial value to the lower bound value ($\varepsilon_{\text{p}} = 1 \times 10^{-5}$). This circumstance means that after CCN routers start caching data packets, the cache-hit ratio continuously increases. Hence, the caching algorithm gradually reduces f_{add}. In state B, the base layer caching probability (P_0) takes 15 s to decrease to the steady-state of the cache, which is defined by a probability lower than 0.001. According to the data priorities, the caching probability of the first enhancement layer (P_1) is less than P_0 and rapidly falls, as does f_{add}. The steady-state C continues until the requested contents are totally changed at the 2,000-s mark of the simulation. The caching algorithm resets f_{add} to the initial-state, which is equal to 1, directly affecting the caching probability of every layer. The new initial-state is shown in state D. Then, f_{add} repeatedly falls to ε_{p} while P_0 and P_1 converge to steady-state E and F respectively as shown in Fig. 5.

The characteristics of the rightmost CCN router are shown in Fig. 6 when the CS capacity is equal to ten

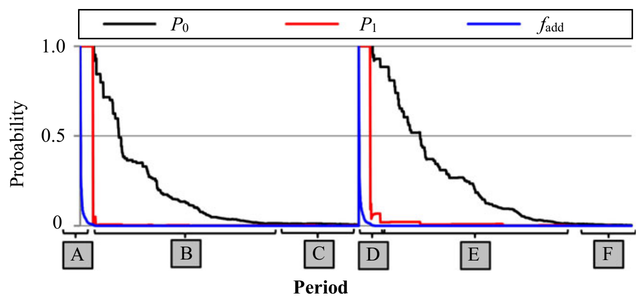

Fig. 5. Rightmost CCN router instantaneous characteristics of caching algorithm components, when the cache capacity is equal to 1% of content population.

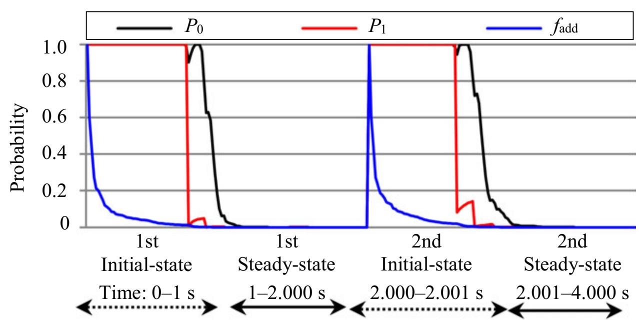

Fig. 6. Rightmost CCN router instantaneous characteristics of caching algorithm components, when the cache capacity is equal to 10% of content population.

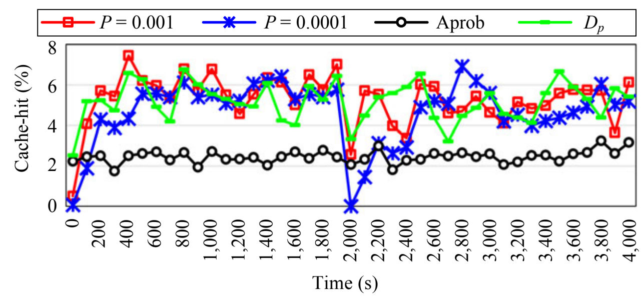

Fig. 7. Rightmost CCN router cache-hit percentage characteristics when the cache capacity is equal to 1% of content population.

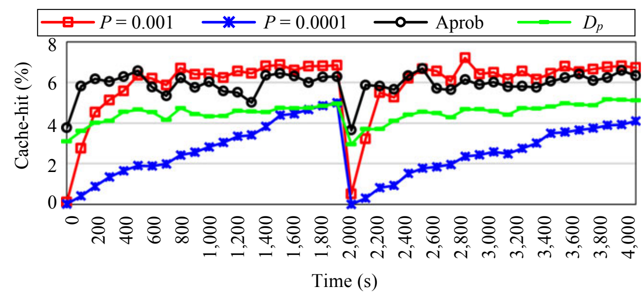

Fig. 8. Rightmost CCN router cache-hit percentage characteristic when cache capacity is equal to 10% of content population.

percent of content population. The huge capacity of the CS makes slight changes to the characteristics of caching algorithm components. First, f_{add} takes less than one second to reduce to ε_p, while the caching probabilities of the base layer and the first enhancement layer in the first initial-state converge to the steady-state within one second. Therefore, the time to converge to the first steady-state is longer than it is when the CS capacity is equal to one percent. f_{add} is reset to the second initial-state due to the changes of content popularity. The characteristics of the second initial-state and steady-state are the same as during the first 2,000 s.

The cache-hit percentage characteristics of the rightmost CCN router when the CS capacities are equal to one percent and ten percent of content population are shown in Figs. 7 and 8. To reduce the variance of the instantaneous cache-hit percentage, we determine the cache-hit characteristic by computing the average cache-hit percentage within the 20-second time window. For both sizes of CS capacity, the cache characteristics are almost the same. The cache-hit percentage is increasing in the initial-state and converging to the steady-state.

In Fig. 7, the proposed caching algorithm takes only 20 s to bring the cache-hit to the steady-state while Prob spends 400 s. This shows that our caching algorithm satisfies the requester within a shorter time. Considering the change of the requested content when the simulation time is equal to 2,000 s, the proposed algorithm is more stable than Prob, as its cache-hit has lower variance.

When the cache capacity is equal to ten percent of content population, cache-hit characteristics of the rightmost CCN router shows that Prob require longer initial time to reach the cache steady-state. When the caching probability is equal to 0.001, although this provides a high cache-hit percentage, it takes 200 s to reach the cache-hit percentage level of the proposed scheme and approximately 400 s to converge to the steady-state. If the content requester requests totally new content, and φ^t is equal to zero, the cache-hit percentage of the proposed algorithm does not drop to zero and does not need to take as much time to reach a steady-state as do Prob.

b. Experimental Results

The experimental results of the cascading network topology when the cache capacity is equal to one percent and ten percent are shown in Tables 1 and 2, respectively.

Table 1. Cascading topology experimental results where bold values refer to the best result. (CS = 1%).

Algorithm	Content	V	Cache-hit (%)	Server load (Packets/s)	Network traffic (Packets/s)
P = 1	(2, 5)	0.237	1.994	912.39	813.26
	(2, 3)	0.237	1.994	547.43	487.96
Average		0.237	1.994	729.91	650.61
P = 0.1	(2, 5)	0.252	2.042	912.15	813.92
	(2, 3)	0.252	2.041	547.26	488.35
Average		0.252	2.0415	729.71	651.135
P = 0.01	(2, 5)	0.289	3.097	873.16	804.26
	(2, 3)	0.286	2.927	527.54	483.79
Average		0.2875	3.012	700.35	644.03
P = 0.001	(2, 5)	0.303	**4.062**	**837.98**	**792.79**
	(2, 3)	0.297	**3.689**	**510.68**	**478.35**
Average		0.3000	**3.8755**	**674.33**	**635.57**
P = 0.0001	(2, 5)	0.301	4.013	838.97	794.69
	(2, 3)	0.292	3.451	515.49	480.74
Average		0.2965	3.732	677.23	637.58
Aprob	(2, 5)	0.225	2.241	906.15	812.03
	(2, 3)	0.258	2.213	543.10	487.11
Average		0.242	2.227	724.63	649.57
Proposed	(2, 5)	**0.331**	3.679	852.31	796.04
	(2, 3)	**0.325**	3.416	517.04	479.54
Average		**0.328**	3.5475	684.675	637.79

Table 2. Cascading topology experimental results where bold values refer to the best result. (CS = 10%).

Algorithm	Content	V	Cache-hit (%)	Server load (Packets/s)	Network traffic (Packets/s)
P = 1	(2, 5)	0.297	9.249	666.06	685.66
	(2, 3)	0.297	9.249	399.63	411.40
Average		0.297	9.249	532.85	548.53
P = 0.1	(2, 5)	0.509	16.233	450.57	599.51
	(2, 3)	0.511	16.225	270.53	360.00
Average		0.510	16.229	360.55	479.76
P = 0.01	(2, 5)	0.565	**20.180**	**351.21**	**550.79**
	(2, 3)	0.564	**20.077**	**212.22**	**331.11**
Average		0.5645	**20.129**	**281.715**	**440.95**
P = 0.001	(2, 5)	0.566	20.149	353.67	552.39
	(2, 3)	0.564	19.977	214.25	332.48
Average		0.565	20.063	283.96	442.44
P = 0.0001	(2, 5)	0.429	11.274	594.39	683.64
	(2, 3)	0.432	11.311	355.75	441.78
Average		0.4305	11.293	475.07	562.71
Aprob	(2, 5)	0.562	20.096	361.42	557.72
	(2, 3)	0.562	19.810	212.95	333.12
Average		0.562	19.953	287.19	445.42
Proposed	(2, 5)	**0.655**	18.887	383.30	583.05
	(2, 3)	**0.671**	18.707	236.16	349.96
Average		**0.663**	18.797	309.73	466.51

First, we consider the results when the cache size is equal to one percent of population, which are in Table 1. While Prob (P = 0.001 and P = 0.0001) yield better caching performance and network performance, the proposed caching algorithm yields the best data quality to the requester. Even though the caching algorithm cannot achieve a network performance as high as Prob, its network performance is comparable. The caching algorithm gives approximately a ten percent improvement in data quality compared with Prob.

Table 2 shows the results when the cache capacity is equal to ten percent of the content population. The results have the same tendency as those of the one percent of content population. It yields the best received quality for the requester with network performance comparable to Prob.

When the cache capacity is increased to be equal to ten percent of content population, Prob with P = 0.01 and P = 0.001 offer the best cache-hit percentage in the rightmost router. Although the rightmost router of D_p cannot provide the best caching operation, the majority of routers in the cache network (R2, R3, and R4) yield better cache-hit percentages.

Figures 9 and 10 give the caching performance of CCN routers using the average cache-hit percentage of each CCN router. When the cache capacity is equal to one percent of population, Prob with P = 0.001 offers the highest cache-hit percentage except in R1 and R2, whereas D_p performs as well as Prob with its best caching probability.

Even if Prob and Aprob offer a higher cache-hit ratio than D_p, D_p provides a higher data quality to the receivers. Due to caching more important data packets, its successful

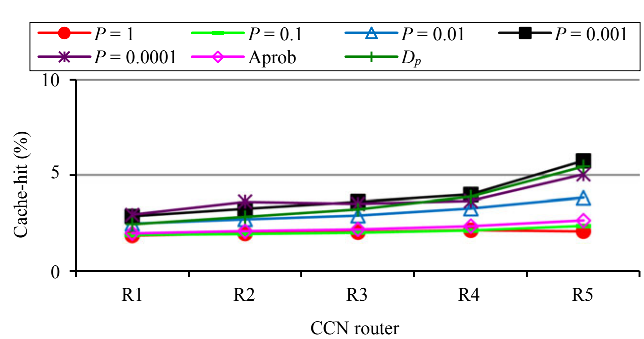

Fig. 9. Average cache-hit percentage of each CCN router when cache capacity is equal to one percent of content population.

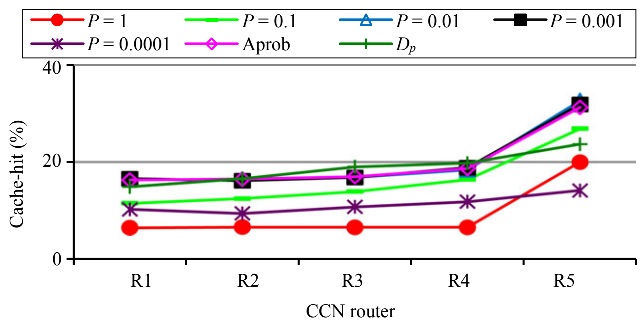

Fig. 10. Average cache-hit percentage of each CCN router when cache capacity is equal to ten percent of content population.

data reconstruction rate is higher than that of the other algorithms. This implies that the high cache-hit ratio does not mean that the cache-hit content is useful to the content reconstruction process.

B. Random Topologies

This section evaluates the proposed adaptive results when applied to randomly generated networks with various routing algorithms. We utilize a universal caching scheme with the caching probability equal to one and Prob with the caching probability equal to 0.001 as our benchmarks. We select Prob with the caching probability equal to 0.001 because it gives the best performance compared with other caching probability. We generate 30 random topologies and compute the average performance from requesters as well as the networks from these 30 random topologies.

The results in Table 3 show that D_p delivers better data quality to the requesters than Prob. Conversely, Prob offers higher performance than D_p for other metrics. Regarding the routing protocols, D_p works well with every routing protocol we consider in the evaluation. When D_p is used with SP, it offers the best data quality to the requester. When Prob is used with OCRP, fewer interest packets arrive at the content provider. With the same routing protocol, the results are similar to the results for the cascading network topology.

Table 3. Average improvement gain (%) of random topologies compared with the universal caching scheme where bold values refer to the best cache-hit gain.

Algorithm	Content	V	Cache-hit ratio	Server load	Network traffic
P = 0.001	SP	12.060	9.851	**–18.613**	**–11.215**
	CP	13.860	**9.877**	–17.087	–7.771
	OCRP	**15.221**	9.530	–17.290	–7.771
Aprob	SP	10.390	8.082	**–12.059**	**–7.681**
	CP	12.364	**8.309**	–10.947	–5.109
	OCRP	**13.754**	8.038	–11.412	–5.046
D_p	SP	20.268	6.089	**–12.031**	**–5.411**
	CP	22.281	**6.179**	–11.469	–3.034
	OCRP	**24.071**	6.087	–11.829	–0.2979

When we use the caching algorithm with OCRP, the data quality is improved from universal caching scheme with OCRP by 24.071%. If Prob is implemented with CP, it offers a 9.877% cache-hit percentage gain, which is the best improvement among all scenarios. The cache-hit percentage gain of D_p with CP is better than that of other routing protocols. However, Prob with SP yields the highest reduction rate in server load and network traffic. The results show that the performance of the caching schemes is independent of and not affected by the routing protocol.

VI. Conclusion

A dynamic probabilistic caching algorithm with content priorities for CCN was presented to offer a better reconstructed data quality to the requesters while maintaining network performance at the same level as the previous probabilistic caching algorithm. The caching probability was calculated using the priorities of data contents, a multiplication factor, and an addition factor. These variables were adjusted along with the time-varying content and network environments. We evaluated the performance of the caching algorithm compared with the previous caching schemes by computer simulation under cascading and random network topologies. The experimental results were described in terms of successful data reconstruction rate and network performance. The results showed that the proposed algorithm offered higher data quality to the requester than the existing caching schemes while maintaining acceptable network performance.

Acknowledgements

This work was supported by the Research Strengthening Project of the Faculty of Engineering, King Mongkut's University of Technology Thonburi, Bangkok, Thailand.

References

[1] V. Jacobson et al., "Networking Named Content," *Int. Conf. Emerg. Netw. Experiments Technol.*, Rome, Italy, Dec. 1–4, 2009, pp. 1–12.

[2] L. Zhang et al., "Named Data Networking," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, July 2014, pp. 66–73.

[3] G. Xylomenos et al., "A Survey of Information-Centric Networking," *IEEE Commun. Surveys Tutorials*, vol. 16, no. 2, July 2013, pp. 1024–1049.

[4] K. Cho et al., "Wave: Popularity-Based and Collaborative In-network Caching for Content-Oriented Networks," *IEEE Conf. Comput. Commun. Workshop*, Orlando, FL, USA, Mar. 25–30, 2012, pp. 316–321.

[5] K. Suksomboon et al., "Popcache: Cache More or Less Based on Content Popularity for Information-Centric Networking," *IEEE Conf. Local Comput. Netw.*, Sydney, Australia, Oct. 21–24, 2013, pp. 236–243.

[6] A.A. Rocha et al., "DSCA: A Data Stream Caching Algorithm," *Proc. Workshop Content Caching Delivery Wireless Netw.*, Heidelberg, Germany, Dec. 1–4, 2016, pp. 1–6.

[7] Y. Wang et al., "Design and Evaluation of the Optimal Cache Allocation for Content-Centric Networking," *IEEE Trans. Comput.*, vol. 65, no. 1, Jan. 2016, pp. 95–107.

[8] K. Mokhtarian and H.A. Jacobsen, "Flexible Caching Algorithms for Video Content Distribution Networks," *IEEE/ACM Trans. Netw.*, vol. 25, no. 2, Apr. 2017, pp. 1062–1075.

[9] W. Zhang et al., "Energy-Aware Caching," *2015 IEEE 21st Int. Conf. Parallel Distrib. Syst.*, Melbourne, Australia, Dec. 14–17, 2015, pp. 473–480.

[10] W. Zhang, F. Liu, and R. Fan, "Cache Matching: Thread Scheduling to Maximize Data Reuse," *Proc. High Performance Comput. Symp.*, San Diego, CA, USA, Apr. 13–16, 2014, pp. 1–8.

[11] K. Kvaternik et al., "A Methodology for the Design of Self-Optimizing, Decentralized Content-Caching Strategies," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, Oct. 2016, pp. 2634–2647.

[12] Y. Wang, M. Xu, and Z. Feng, "Hop-Based Probabilistic Caching for Information-Centric Networks," *IEEE Global Commun. Conf.*, Atlanta, GA, USA, Dec. 9–13, 2013, pp. 2102–2107.

[13] S. Tarnoi et al., "Performance of Probabilistic Caching and Cache Replacement Policies for Content-Centric Networks," *Annu. IEEE Conf. Local Comput. Netw.*, Edmonton, Canada, Sept. 8–11, 2014, pp. 99–106.

[14] N. Laoutaris, H. Che, and I. Stavrakakis, "The LCD Interconnection of LRU Caches and Its Analysis," *Perform. Eval.*, vol. 63, no. 7, July 2006, pp. 609–634.

[15] W.K. Chai et al., "Cache Less for More in Information-Centric Networks (extended version)," *Comput. Commun*, vol. 36, no. 7, Apr. 2013, pp. 758–770.

[16] I. Psaras, W.K. Chai, and G. Pavlou, "Probabilistic In-Network Caching for Information-Centric Networks," *Proc. ACM SIGCOMM Workshop Inform.-Centric Netw.*, Helsinki, Finland, Aug. 17, 2012, pp. 55–60.

[17] I. Psaras, W.K. Chai, and G. Pavlou, "In-network Cache Management and Resource Allocation for Information-Centric Networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 11, Dec. 2013, pp. 2920–2931.

[18] W. Sirichotedumrong et al., "Prioritized Probabilistic Caching Algorithm in Content Centric Networks," *Proc. Int. Conf. Comput. Inform. Technol.*, Khon Kaen, Thailand, July 7–8, 2016, pp. 255–265.

[19] S. Tarnoi, V. Suppakitpaisarn, and Y. Ji, "Adaptive Probabilistic Caching Scheme for Information-Centric Networking," IEICE, Japan, Tech. Rep. IEICE-IN2015-51, Sept. 2015.

[20] K. Poularakis et al., "Caching and Operator Cooperation Policies for Layered Video Content Delivery," *IEEE INFOCOM Annu. IEEE Int. Conf. on Comput. Commun.*, San Francisco, CA, USA, Apr. 10–14, 2016, pp. 1–9.

[21] J. Lee, K. Lim, and C. Yoo, "Cache Replacement Strategies for Scalable Video Streaming in CCN," *Asia-Pacific Conf. Commun.*, Bali, Indonesia, Aug. 29–31, 2013, pp. 184–189.

[22] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN simulator for NS-3," NDN, Tech. Rep. NDN-0005, 2012.

[23] S. Mastorakis et al., "ndnSIM 2.0: A New Version of the NDN Simulator for NS-3," Tech. Rep. on NDN, NDN-0028, 2015.

[24] R.M. Perea, "The Media Plane," *Internet Multimedia Communications Using SIP*, San Francisco, USA: Morgan-Kaufmann, 2008, p. 203.

[25] L. Wang et al., "OSPFN: An OSPF Based Routing Protocol for Named Data Networking", NDN Tech. Rep. TR003-OSPFN, July 2012.

[26] S. Tarnoi et al., "Cooperative Routing Protocol for Content-Centric Networking," *Annu. IEEE Conf. on Local Comput. Netw.*, Sydney, Australia, Oct. 21–24, 2013, pp. 716–719.

[27] S. Tarnoi, W. Kumwilaisak, and Y. Ji, "Optimal Cooperative Routing Protocol Based on Prefix Popularity for Content Centric Networking," *Annu. IEEE Conf. Local Comput. Netw.*, Edmonton, Canada, Sept. 8–11, 2014, pp. 414–417.

Warit Sirichotedumrong received his BE in electrical communication and electronic engineering from King Mongkut's University of Technology Thonburi, Thailand in 2014. He obtained his M.E. in electrical engineering from the same university in 2017. He received the best paper awards from the 12th International Conference on Computing and Information Technology, and the 39th Thailand Electrical Engineering Conference in 2016. From November 2016 to April 2017, He was a software engineer at AmiVoice Thai Co., Ltd., Bangkok, Thailand. Currently, He is a research assistant at King Mongkut's University of Technology Thonburi, the Electronics and Telecommunication Department, Bangkok, Thailand. His research interests include algorithmic design and optimization for multimedia communication and computer networks.

Wuttipong Kumwilaisak obtained his BE degree from Chulalongkorn University, Thailand in 1995, and his MS and PhD degrees from University of Southern California, Los Angeles, CA, USA in 2003, all in electrical engineering with support from the Thai Government Scholarship. From May to August of 2001 and 2002, he was a research intern at Ericsson Eurolab, Aachen, Germany, and at Microsoft Research Asia, Beijing, China, respectively. From April 2003 -August 2004, he was a senior engineer and project leader of the mobile platform solution team and multimedia laboratory at Samsung Electronics, Suwon, Korea. He was a postdoc fellow at Thomson Research Laboratory, Princeton, USA from March to November 2006. Currently, he is an Associate Professor at King Mongkut's University of Technology Thonburi, in the Electronics and Telecommunication Department, Bangkok, Thailand. Wuttipong's research interests are in optimization and algorithmic design for wireless communications, multimedia processing, and multimedia communication systems. He received the best paper award from the multimedia communication Technical Committee of the IEEE Communication Society in 2005, the best paper awards from the 12th International Conference on Computing and Information Technology, and the 39th Thailand Electrical Engineering Conference in 2016.

Saran Tarnoi received his BE degree in electronics and telecommunication engineering from King Mongkut's University of Technology Thonburi, Thailand in 2010. He received ME in electrical engineering from the same university in 2012. He obtained his PhD degree from Department of Informatics of the Graduate University for Advanced Studies (SOKENDAI), Miura, Japan, in 2015. He spent several years working on addressing networking and multimedia transmission problems with content-centric networking, optimization, and network coding. He received the best paper awards from the 38th IEEE Local Computer Network Conference in 2013, the 12th International Conference on Computing and Information Technology, and the 39th Thailand Electrical Engineering Conference in 2016. He is currently interested in applying theoretical solutions to practical problems in manufacturing and business processes. He is an associate manager of the IT department at CPPC Public Co., Ltd, Bangkok, Thailand.

Nattanun Thatphithakkul received his BE and ME degrees from Suranaree University, Thailand in 2000 and 2002, respectively. He received his PhD in computer engineering from King Mongkut's Institute of Technology Ladkrabang, Thailand in 2008. He received the best paper awards from the 12th International Conference on Computing and Information Technology, and the 39th Thailand Electrical Engineering Conference in 2016. He is now Chief of the Accessible Innovation and Universal Design Laboratory of the National Electronics and Computer Technology Center, Pathum Thani, Thailand. His research interests include speech and speaker recognition, speech synthesis, natural language processing, human-machine interaction, and assistive technology.

https://doi.org/10.4218/etrij.17.0117.0117