A Novel Decision-Making Mechanism of Resource Scheduling in High-Speed Trains in Push-based Wireless Converged Broadcasting and Cellular Networks

Bing Li*, Jian Xiong†, IEEE Member, Bo Liu†, IEEE Member, Lin Gui†, IEEE Member
and Meikang Qiu‡, Senior IEEE Member

*†Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China
‡Department of Computer Science, Pace University, New York City, USA.
*Email: libing0826@hotmail.com
†Email: {xjarrow, liubo_lb, guilin}@sjtu.edu.cn
‡Email: mqiu@pace.edu

***Abstract*—Int this paper, we propose a novel decision-making mechanism of Resource Scheduling (RS) in high-speed trains in the push-based Wireless Converged Broadcast and Cellular Network (WCBCN) . In this model, we transform dynamic programming problem into the static issue. When the train set out from the origin station, the WCBCN start to broadcast the hot-spot information which have been already downloaded at the Cloud Computing Center (CCC) to all passengers in the trains.Because there will be some passengers on or off the train in the next railway station. In order to maximize the capacity of the WCBCN, the CCC will decide whether to retransfer the most popular contents or not before the train set out from the next station. It's intuitive that the capacity will improve to some extent. Simulations show that proposed model for the high-speed mobile environment can increase the capacity by 50%-60% comparing with the traditional one.**

***Index Terms*—Wireless Converged Broadcast and Cellular Network (CWBCN), Cloud Computing Center (CCC), High-speed train, information retransmission.**

I. Introduction

Recently, more and more people choose the high-speed railway as their means of transportation. due to their Low-Carbon and time saving. Moreover, high-speed trains with a speed of 300 KM/h have been rapidly deployed all over the world [1]. The growing consumption of multimedia content and the development of big data bring a significant burden of mobile cellular network's data traffic, limited network capacity and severe request congestion would predictably be the key bottlenecks. With the data explosion and the demand of various multimedia content increasing, cellular network burden continues to grow, especially with the smart phones are popularized, mobile data traffic continue to climb at a more rapid rate than they have been over the past few years. It seems that the pace of existing cellular network infrastructure development can not meet the needs of the exponential growth of wireless data traffic burden [**?**].

Under this situation, converged network, as single wireless network could no longer meet the demand of many scenarios, has become a focus recently. Previously a number of topics have been researched in the field of converged network, like Converged Broadcasting and Cellular Networks *(CBCN)* based on DVB-H/DVB-T, DMB, ISDB-T, MediaFLO, etc, and in the undergoing research of 5G cellular technologies, Mobile broadcasting comes of age in LTE eMBMS(evolved Multimedia Broadcast Multicast Services), eMBMS Provides Excellent Broadcast Performance, at present the application of LTE-broadcast(LTE-eMBMS) is with an efficient P2M(point to multi-point) distribution feature that can simultaneously serve many LTE-broadcast capable terminals with the same content. Based on this, LTE-broadcast can be used to boost network capacity for live or on-demand content such as top websites, breaking news or popular video which can be broadcast-offloading the network and providing users with a superior experience [**?**].

However, under this circumstances, the existing application of converged network is more of a real time solution, which means the users are requesting for the same content at the same time. Based on statistical user trace, the massive requests are not randomly consumed on all contents. A number of researches have been done around the user request distribution [**?**], indicating that there exists Matthew effect in those massive requested multimedia contents, and helping some contents gain popularity [**?**]. That is, a majority number of users are mainly requesting for a small part of contents, we name these contents *popular contents*. To extend the limitation, we proposed a cache mechanism in the client (*UE*) to cache the most popular contents so that equivalently the application can handle with users' requests asynchronously. Caching is widely used to improve user experience, alleviate request congestion and reduce resource consumption in communication networks. In the context of wireless communication networks, various

978-1-4673-9044-6/16/$31.00 ©2016 IEEE

caching schemes have been proposed, including base station caching, proxy caching, relay caching and client caching [?]. In [?], the authors consider a wireless caching network where all the BSs are connected to a central controller via backhaul links. In [?], the authors studied video caching in the radio access network and proposed caching policies based on the user preference profile.

While most of them take base station cache or relay cache more into account in existing researches, as most of user terminals have been updated to smartphones, and the local data storage have been lower-cost and faster-response, there are reasons to believe locally forward caching in user equipments would be a practicable solution. "Local Forward Caching" refers to a terminal saving a content forwardly, then reusing it later instead of re-requesting the same asset from the base station again.

In this paper, to support the popular contents forward caching based on converged network, it is required to propose a content-pushing scheme and an efficient hybrid scheduling algorithm on *UE* side and evaluate its performance.

The main contributions of this work can be summarized as follows:

- Firstly, we introduce a UE-based cache replacement mechanism, in which we take both frequency and recency of popular services into consideration. Simulations show that the adopted algorithm in our architecture can significantly improve the capacity of the CWBCN.
- Then, based on the cache replacement mechanism, a sleep-awake algorithm of pushing process is proposed to reduce the energy consumption of UE. Simulations show that the performance of the proposed sleep-awake algorithm is further improved and far outweighs the traditional ones.

The rest of the paper is organized as follows. In Section II, we discuss our system model which includes the analysis of the popularity distribution of user request, architecture of request and response scheme. Section III presents our problem formulation and the proposed algorithm. In Section IV, we evaluate our work by performance analysis. Section V concludes the paper.

II. System model

A. User Request Distribution

Several research contributions on meeting the distribution of the popularity of requesting contents, like user generated content, video-on-demand [?]. Research shows that ZIPF's law provides an statistically accurate fit with these multimedia contents [?]. To confirm the validity, we mine data on the real Internet traffic traces from one of the most popular Chinese portal sites, and the result is consistent with the zipf law.

B. Converged Network with Popular Contents Forward Push

Consider a single cell in a traditional cellular network, there are N active users in the coverage, all of users send request for the contents to the same base station. For illustrative purposes, here gives a simple example, in a time slot, each user i (u_i) send a request for content j (c_j), let us denote the bandwidth of each subchannel as B_{csi}, the average bandwidth efficiency as B_{cei} and assume there is no blocked request, as a result, the network capacity is $\sum_{i=1}^{N} B_{csi}B_{cei}(bps/Hz)$.

In the converged network with popular contents forward push, a popular contents set $pc = \{m_1, m_2, m_3, ..., m_k\}$ are forward pushed to each user through broadcasting, and locally cached by each user. Once the requested content $c_j \in pc$, we name it a cache hit event, and if the hit rate is r_i, that is, in this time slot, there are $r_i\lambda_i$ cache events occur, let us denote the broadcast bandwidth as B_b, and the average bandwidth efficiency as B_{ce}, then the equivalent network capacity is $\sum_{i=1}^{N} B_{csi}B_{cei} + r_i\lambda_i B_b B_{be}$. This is the prototype of our ideas.

C. Base Station Controller Server (BSCS)

Web proxies and Content Distribution Networks are widely used to accelerate web content delivery and to conserve internet bandwidth. In our architecture, we add a layer between UE and basestation, called Base Station Controller Server, as a proxy server. Traditional proxy servers are deployed by Internet Service Providers (*ISPs*) to deal with increased Web traffic and optimize the content delivery on the Web. Caching is widely used to improve user experience. ISPs use proxies to store the most frequently or most recently requested content [?]. We innovatively proposed *BSCS* to dynamically store and schedule the most popular contents for the UEs in the base station coverage area. The *BSCS* preloads contents from *ISPs* into its memory based on a set of predetermined popularity rank for future broadcast pushing.

D. Scenario

Fig. 1 is the scenario of this paper, in this architecture, on the station side, in addition to traditional cellular base station, there is a base station which serves multi-cell simultaneously by a *BSCS*, it can automatically schedule its cache with *ISP* and update the cache with upcoming popular contents. In the meantime, it polls popular contents to all the users in its region by broadcasting. On the user side, each *UE* has its own cache for the broadcasting popular contents, to increase the flexibility and provide better scheme evolution, each *UE*'s broadcasting channel can switch on or off autonomously.

The request stream can be described as Fig 2: When a UE requests an Internet service, such as HTTP, the client checks its local cache for the data and, if the data is available, sends it to the client immediately. If the data is not available, the client requests the data from the hierarchical BSCS then base station requests to the origin ISP, and then returns the data to the client.

III. Problem Formulation and Algorithm

In this section, basic definitions and key notations are first introduced, and provide a basic formulated model. Then the UE-based cache replacement algorithm and push strategy are described.

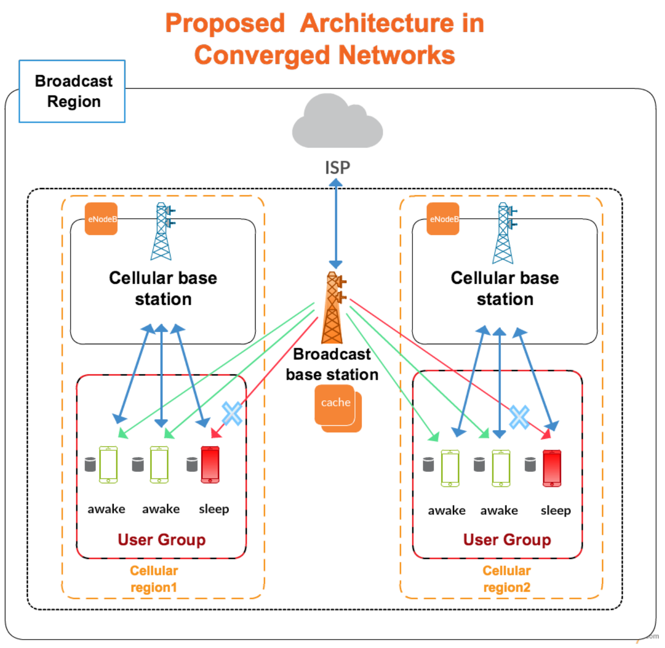


Fig. 1: Proposed converged networks architecture

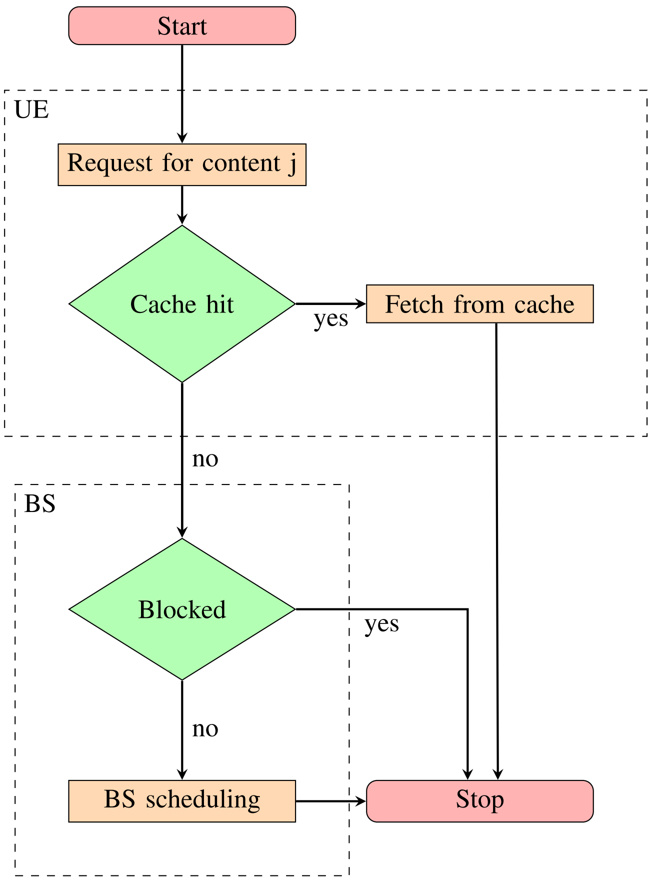


Fig. 2: BS to UE request stream

A. Problem Formulation

The definitions of parameters shown in Table I.

TABLE I: definition of Parameters

Parameter	Definition
B_b	Broadcasting bandwidth
B_{be}	Broadcasting bandwidth efficiency
B_{cs}	Cellular subchannel bandwidth
B_{ce}	Cellular subchannel bandwidth efficiency
N	Number of users in one cell
N_s	Num. of subchannels
N_b	Num. of broadcasting contents
N_{bmax}	Ceiling of broadcasting contents
p_{ci}	Power consumption on cellular network
p_{bi}	Power consumption on broadcasting network
r_i	Broadcasting hit rate of user i's requests
ξ_i	User i cache size
ζ_0	Average size of content
λ_i	User i requesting rate
η	Throughput Efficiency
P	Allowed max power of a user for data transmission

As described above, user request distribution follows zipf distribution, the probability density function of the zipf distribution is given by

$$pdf : f(j; z, N) = \frac{(1/j)^z}{\sum_{n=1}^{N}(1/n)^z} = \frac{1}{j^z H_{N,z}} \tag{1}$$

For evaluating system performance, we define aggregate throughput of the converged network as:

$$\begin{aligned} TH &= TH_b + TH_c \\ &= TH_b + \sum_{i}^{N} TH_{ci} \\ &= N_b\zeta_0 + \sum_{i}^{N}(1-r_i)\lambda_i\zeta_0 \end{aligned} \tag{2}$$

where TH_b is throughput of all contents which is pushed in broadcast, and TH_{ci} is the user i's cellular network throughput.

Therefore, the Throughput Efficiency is:

$$\begin{aligned} \eta &= \frac{\textit{effective throughput}}{\textit{aggregate throughput}} \\ &= \frac{\sum_i^N \lambda_i\zeta_0}{TH} \\ &= \frac{\sum_i^N \lambda_i\zeta_0}{N_b\zeta_0 + \sum_i^N (1-r_i)\lambda_i\zeta_0} \end{aligned} \tag{3}$$

And as discussed in section II.B, the equivalent network capacity is:

$$\sum_{i=1}^{N} B_{csi}B_{cei} + r_i\lambda_i B_b B_{be} \tag{4}$$

For simplicity, we ignore the difference of cellular subchannel bandwidth of each user, so the problem can be written as:

$$\max NB_{cs}B_{ce} + \sum_{i=1}^{N} r_i\lambda_i B_b B_{be} \tag{5}$$

Subject to

$$r_i = \sum_{n=1}^{\xi_i} f(j;z;n) \tag{6}$$

Considering that our work is based on a converged broadcast and cellular networks, for evaluate the user experience, it is necessary to point out the difference of power consumption between broadcast and cellular networks.

For simplicity, let the UE's power consumption of a request for cellular network given by p_c, and the power consumption of pre-caching a broadcast content item is p_b, then for the user i, its power consumption can be given by:

$$P_i(\xi_i, r_i) = p_b\xi_i + \lambda_i(1 - r_i)p_c \tag{7}$$

Minimizing the power consumption of all users can be described as:

$$\min \sum_{i\ =1}^{N} (P_i(\xi_i, r_i)) = \min \sum_{i=1}^{N} (p_b\xi_i + \lambda_i(1 - r_i)p_c) \tag{8}$$

Subject to

$$r_i = \sum_{n=1}^{\xi_i} f(j;z;n) \tag{9}$$

It is an optimization problem, the hit rate has close relationship with the cache size, and the related replacement algorithm. Considering the difference of user behavior, the λ_i follows a normal distribution. In next section, based on our push scheme, we will get a simulation-based conclusion.

The algorithms we will introduce focus on two main processes: pushing process and caching process. In the pushing process, our goal is to cut down the deadweight loss of energy consumption by UE adaptively adjustment. In the cache process, we develop the algorithms to increase the hit rate so that improve the system performance. In the following we will describe these two processes in details.

B. Cache Replacement Algorithm to Improve capacity:

Before we introduce the pushing process, it is necessary to introduce the related caching process, there is a rich body of the existing papers on several aspects of caching strategy including object replacements, cooperation cache in traditional wired networks [**?**], the scenario this paper explored is mainly based on cellular network, in which the nodes are mainly mobile equipment, therefore it is different from wired networks in physical layer or other upper layer like application layer. As a result, the available cache solutions in existing wired network are not directly applicable for the proposed converged architecture.

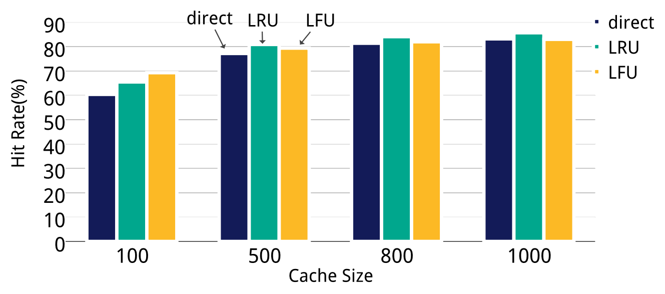
Most of the cache strategies are based on two key dimensions: *recency* and *frequency*, in this paper, we focus on the popularity of each requested content, the *frequency* could well represent the *popularity*. In addition, Another significant feature of popular contents is timeliness, this falls under the category of *recency*. LRU (Least Recently Used) and LFU (Least Frequently Used) are two typical cache replacement policy, respectively. As we talk about *popular contents*, both of the requesting frequency and contents timeliness should be concerned about. And because of this, in this paper, we adopt two UE-based LRU and LFU algorithms based on our converged network architecture, which are given in Algorithm 1.

Algorithm 1 Related cache replacement algorithm

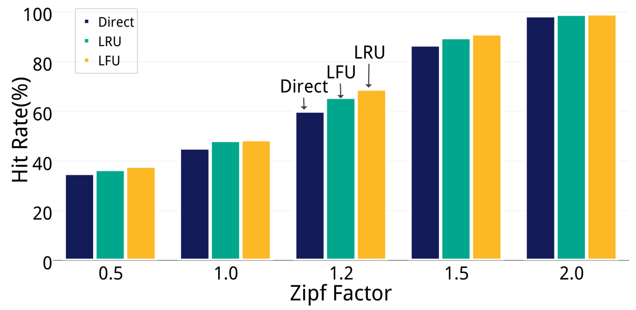
```
1:  procedure UE-BASED LRU
2:      for each request do
3:          if $\exists i$ : $cache[i] ==$ requested content then
4:              $cache[i].count = 0$
5:          else
6:              if cache.length$>N_b$ then
7:                  $t = \arg\max_p\{cache[].count\}$
8:                  $cache[t] \overset{replace}{\longleftarrow}$ requesting content
9:              else
10:                 $cache \overset{add}{\longleftarrow}$ requesting content
11:             end if
12:         end if
13:         increment all the counts
14:     end for
15: end procedure
16: procedure UE-BASED LFU
17:     for each request do
18:         if $\exists i$ : $cache[i] ==$ requested content then
19:             $cache[i].count+ = 1$
20:         else
21:             if cache.length$>N_b$ then
22:                 $t = \arg\min_p\{cache[].count\}$
23:                 $cache[t] \overset{replace}{\longleftarrow}$ requesting content
24:             else
25:                 $cache \overset{add}{\longleftarrow}$ requesting content
26:             end if
27:         end if
28:     end for
29: end procedure
```

C. Pushing Strategy to Save Energy:

While in the real-world situation, not every UE is interested in these popular contents, in terms of this paper, as we mentioned above, every UE has its unique hit rate because of personalization, under this situation, for these UEs, pushing and forward caching those unwanted contents would significantly increase the cost of energy and network load. To meet the challenges, in this paper, we are interested in an asynchronous sleep-awake scheduling strategy, the main steps of the strategy are presented in Algorithm 2. In this strategy, each UE wakes up independently depends on its own *cache hit rate* and *request rate* in order to enhance user experience. That is, when the UE's cache mechanism is in a low hit rate or a low request rate, it falls into a sleep state, it closes the



(a) Hit Rate vs Cache Size



(b) Hit Rate vs Zipf Factor

Fig. 3: Hit Rate simulation

broadcast channel to avoid caching unwanted contents, with the help of this, the energy consumption of UEs is effectively reduced. At the same time, the cache mechanism is listening on the hit rate and request rate, once exceeding a threshold value, the UE turns to an awake state. In this state, the UE opens the broadcast channel and start caching the popular services being pushed.

IV. Simulation and Analysis

TABLE II: Simulation Parameters

Parameter	Values
Num. of users	$N = 30$
Cache Size	cache = [100,1000)
zipf factor	$\alpha = 0.5, 1, 1.2, 1.5$
Max. Num. of push contents	$N_b = 1e3$

We simulated the performance of UE's local cache hit rate of the *LRU* and *LFU* cache replace algorithm, for comparison, we also analyzed *direct replace algorithm*, which means cache directly replace the exists one when cache is full. The related parameters are given in the Table II, The numeric simulation result shows in Fig 3. The results show that with proper cache replacement algorithms the UEs' cache efficiency is increased, and makes it a non real time solution, not just applicable with real time scenario like live video. Accordingly the capacity of converged network is further improved compared with the traditional one.

Algorithm 2 proposed sleep-awake scheme

```
1: procedure REQUEST-RESPONSE
2:     for each request from user $i \in N$ do
3:         if $\exists i$ : $ServerCache[i]$ == requested content then
4:             $UserTable[i].HitCount++$
5:         end if
6:         if $UserTable[i].HitCount > PushThreshold$ then
7:             $response.add[PushMode] = Awake$
8:             for each pushed content do
9:                 if content in local cache then
10:                    Continue
11:                else
12:                    fetching and caching
13:                end if
14:            end for
15:        else
16:            $response.add[PushMode] = sleep$
17:        end if
18:    end for
19: end procedure
```

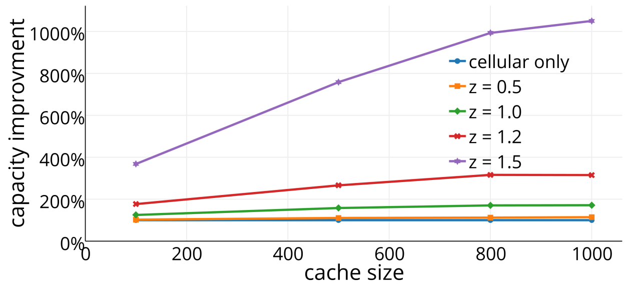


Fig. 4: Capacity improvement comparison by the proposed scheme

Although the proposed converged network architecture brings significant network capacity improvement shown in Fig 4, the inevitable cache miss events would cause the problem of efficiency, so we defined *Throughput Efficiency* given in the equation (3), from the simulation results, we can find a maximum point between 500 to 800(cache size), indicating that the more cache or the more broadcasting is not always better. Too much caching would cause a waste of network resources, and make it lower efficient. For users, this degrades the user experience in battery consumption and storage occupancy.

Based on this phenomenon, we simulate the proposed sleep awake strategy by examine the energy consumption. Fig 5 Fig 6, Fig 7 and Fig 8 are the energy consumption comparison of *CN*, *CBCN* and proposed sleep awake scheme in *CBCN* under the conditions of different zipf factor. By comparison, the converged network turns out to be a more energy-efficient

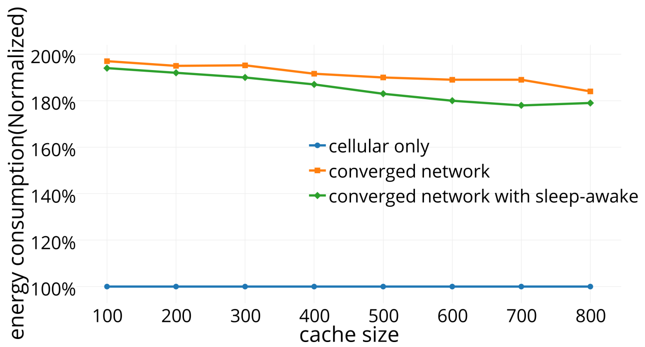


Fig. 5: Energy consumption comparison (zipf factor=0.5)

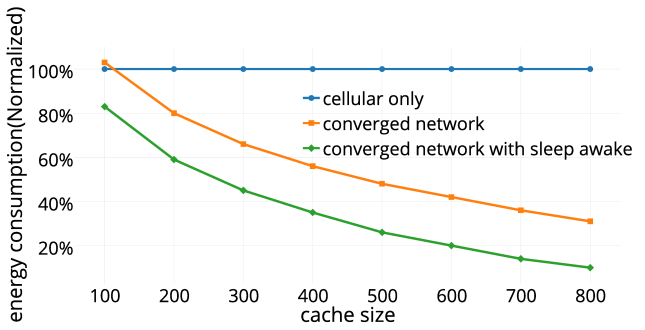


Fig. 6: Energy consumption comparison (zipf factor=1)

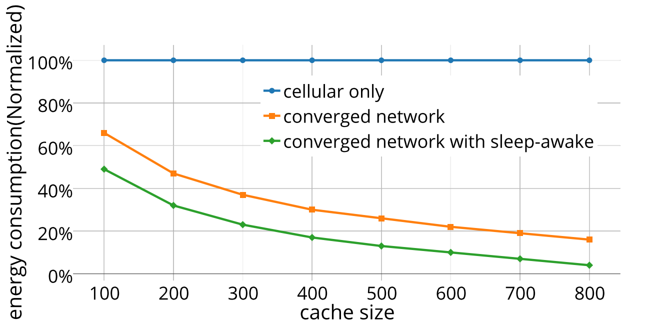


Fig. 7: Energy consumption comparison (zipf factor=1.2)

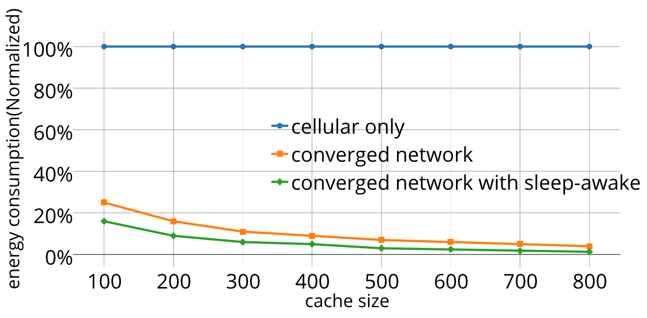


Fig. 8: Energy consumption comparison (zipf factor=1.5)

architecture. Moreover, with the proposed sleep awake pushing scheme, the three figures prove that when the cache contents size of each UE dynamically changes with the different needs, the energy consumption will get a significant improvement.

Compared with traditional cellular network, the energy consumption of converged architecture is reduced by 60% when the cache size is 800 and zipf factor is close to 1, which matches the practice better.

Compared with converged network, the proposed scheme in converged architecture get a 20% more reduction on energy consumption.

In summary, adopting the proposed sleep-awake pushing strategy and cache replace algorithm in our UE-based cache architecture can significantly reduce the energy consumption and improve capacity of *CBCN*. If all the users has a cache and adaptively accept push services, it will save energy consumption by 20%-30% to the converged network.

V. Conclusion

Inspired by the matthew effect of requested contents, we proposed a scheme based on push and forward cache in cellular networks. Towards this direction, we firstly introduce a related push contents cache scheme and cache replace algorithms, which makes it a non real time broadcast solution, and further improve the cache hit rate, thus improve the network capacity of our architecture. And then we suggest a sleep-awake scheme in this paper to decide the proper pushing-throughput of each user, by doing this we reduce the UEs' energy consumption. By comparing our algorithms and scheme with the traditional cellular network and converged network in simulations, it is clear that our scheme can get as much as 60% reduction of energy consumption and nearly 200%(when zipf factor is 1) in capacity improvement compared with cellular only network, and a 20% further improvement compared with converged network. Future work includes real-time pushing scheduling in this scenario and considering more practical and more efficient cache strategy in *CBCN*.

References

[1] Y. Dong, P. Fan, and K. B. Letaief, "High-speed railway wireless communications: Efficiency versus fairness," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 2, pp. 925–930, 2014.