PreFeed: Cloud-Based Content Prefetching of Feed Subscriptions for Mobile Users

Xiaofei Wang and Min Chen, *Senior Member, IEEE*

***Abstract*—While the user demands on mobile multimedia services (e.g., video streaming, web surfing, and news reading) have been souring, the wireless link capacity cannot keep up with the traffic demand. The gap between the traffic demand and the link capacity, along with time-varying link conditions, results in limited quality of mobile services over current 3G/4G cellular networks, such as long loading/buffering time and intermittent disruptions. By utilizing the cloud computing technology, we propose a new framework to improve the quality of Really Simple Syndication (RSS) reading service for mobile users called PreFeed. PreFeed consists of two parts, i.e., cloud-assisted prefetching, which will proactively fetch the multimedia content of the RSS feeds for all subscribed mobile users, and cognitive pushing, which will push the content to mobile users at an appropriate time by evaluating the link quality. Furthermore, the social impact among users in many RSS services is considered. We implement a prototype of the PreFeed framework to evaluate its performance. It is shown that cloud computing can effectively facilitate feed prefetching and cognitive pushing for mobile users; a large portion of traffic load (around 43%–74%) due to redundant downloads can be reduced.**

***Index Terms*—Feed, mobile cloud computing, prefetching, pushing, subscription.**

I. Introduction

DUE TO THE fast development of mobile communication technology, more and more users turn to rely on mobile services, e.g., creating and publishing blogs, reading articles, exploiting interesting figures, and watching video clips. The user demands on mobile services have been significantly souring, but the wireless link capacity cannot keep up with the traffic demand. The gap between the traffic demand and the link capacity, along with time-varying link conditions, results in poor service quality of mobile services over current 3G/4G cellular networks, such as long loading/buffering time and intermittent disruptions [1]. Therefore, the main research directions fall in the following three issues.

1) How to reduce the traffic load via the Internet regarding many mobile users?
2) How to conserve the power energy of mobile devices?
3) How to guarantee the quality of service (QoS) of user demands (i.e., delay) [2].

Manuscript received July 30, 2012; revised December 21, 2012; accepted March 3, 2013. Date of publication November 8, 2013; date of current version February 5, 2014. This work was supported in part by Prof. Min Chen's Youth 1000 Talent Program and in part by the National Natural Science Foundation of China Project under Grant 61300224.

The authors are with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: xiaofeiwang@ieee.org; minchen@ieee.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSYST.2013.2265162

Many current research studies are carried out to improve mobile entertainment services, and many people are still reading news and articles via mobile devices as one of their habits, so-called "mobile reading," while mobile reading is mostly based on the subscription of some information publishers, for example, **seeds**. The seeds will publish specific content related to their topics by Really Simple Syndication (RSS) [3].

RSS is an extensive markup language (XML) application, and the idea behind RSS is to enable users to be informed when the information on the Internet has changed. As the concise RSS formats allow relatively low-bandwidth data gathering even for several different sources, RSS is becoming a mobile technique for notifying the users of new content, particularly in frequently updated websites such as blogs and news portals. For receiving RSS-based content, the RSS feed could contain one or more channels for one RSS document. The channels and items are composed of a title, a description, a URL, and a publishing date, i.e., the abstract of the document. The users can easily access and classify the received contents with the abstract, and then the full version of the update can be further fetched, including multimedia files (e.g., images and videos).

However, while serving RSS feeds with many subscribers, if one user downloads the RSS updates and thus the attached content files, multiple users will induce multiple downloads of the same files via the Internet; hence, there is the significant redundancy downloading problem. In another word, due to the "power law" effect, [4], a large portion of users actually subscribe to only a small amount of RSS feeds, which are very popular publishers, whereas a great number of RSS feeds obtain very limited attention of users [5]. Therefore, in order to avoid unnecessary data delivery from the RSS service providers (SPs) to the clients, it is better to make a "clustering" point with caching functionality to gather RSS updates with content files, and once duplicated requests come, the cached content can be directly utilized without fetching again.

Due to the fast development of cloud computing [6], [7], it becomes easier to deploy centralized virtual servers for large-scale multimedia delivery for mobile users [8]–[10]. In addition, it becomes popular to virtualize private agents in the cloud that are in charge of serving individual users adaptively such as Cloudlets [11] and Stratus [12], which is proved to be more efficient than traditional servers. Therefore, we are motivated to deploy a new framework of RSS service based on cloud computing. With this manner, the RSS feeds of users are collected in the centralized server and duplicated RSS feeds are re-organized and thus filtered out. The cloud server will be automatically fetching the feed content and files and then the delivery to mobile users, whereas duplicated requests can be satisfied directly by the caching in the cloud.

The content delivery to mobile users also faces problems. In a mobile environment, users often suffer from low link quality and thus long downloading (buffering) time. In addition, when the signal is weak, it consumes more battery for transmission. Therefore, recently, the "cognitive" content delivery has become hot, which can automatically monitor user link quality and push the content with a proper rate or at a proper time when the link is good enough to serve the user with high QoS and low battery consumption. It is also proposed to utilize adaptive, or say cognitive, pushing when serving multimedia delivery by cloud computing [9].

Therefore, motivated by the trend of mobile RSS reading and the fast development of cloud computing, in this paper, we propose a framework to improve the quality of RSS service for mobile users, called **PreFeed**, consisting of two parts:

1) **cloud-assisted prefetching**, which will periodically fetch the multimedia content of the RSS feeds for all subscribed mobile users;
2) **cognitive pushing**, which will push the content, or a part of the content, to mobile users at an appropriate time by evaluating the link quality, energy consumption, and user QoS requirements.

We list the features and the corresponding contributions of PreFeed framework as follows.

1) In PreFeed, users first share their subscribed RSS lists to the cloud agent, which will shrink the duplicated RSSs (due to the disparity of the RSS popularity).
2) Depending on user activity profiling, the cloud agent will assign a specific duty cycle for periodically updating the RSS list to obtain new published content with proper interval.
3) From the RSS, after the cloud agent obtains the abstracted information of the content, it will further fetch the content by parsing the XML content and fetch the original multimedia files (mostly from HTML websites) and then store the obtained texts, images, and videos in the server locally.
4) The cloud agent will monitor the link quality and mobile activities of users, as well as the usage condition, so that an appropriate time will be chosen for pushing the content to the mobile devices cognitively.
5) Based on prototype implementation and related evaluation, the PreFeed cloud center can avoid duplicated downloads and reduce the total traffic load up to 74%.

The rest of this paper is organized as follows. We survey related studies in Section II and describe the details of the PreFeed framework in Section III. We discuss optimization on update frequency and cognitive pushing algorithm in Section V. The implementation and evaluation are shown in Section VI, followed by the conclusion in Section VII.

II. Related Work

Different from traditional RSS PC software, the convenient user interface and fast response speed in mobile devices become very important. Furthermore, helping users to filter out unnecessary news and finding interesting ones for mobile users become more critical. Reference [13] studies the recommendation features for an RSS reader, and [14] is also about an intelligent RSS reader for mobile devices by which the users can access to interested content more easily. Google News application [15] also uses online collaborative filtering to find good news for users based on the rating of other users. In addition, many researchers design to push the content (the real file object) to the mobile users based on their RSS subscriptions, e.g., the user preference-based pushing in [16] and the publish–subscribe middleware in [17]. Because of the development of cloud computing technology, pushing by cloud is gaining more attention, e.g., the cloud-based push-styled mobile content dissemination system is shown in [18].

Recently, caching and prefetching have been important and interesting topics for mobile multimedia services. By prefetching, content is always pushed to the device in advance; hence, a user can click to access the content with zero or little downloading delay. The anticipatory retrieval and caching of data for mobile devices in variable-bandwidth environments is studied in [20], and the work in [22] details the design issues of informed mobile prefetching based on the user patterns and link conditions. Similarly, the study in [21] proposes the prefetching scheme focusing on the web content distribution. Prefetching can be effectively carried out also by cloud computing, such as cloud pushing in [19] based on user models, and the AMES-Cloud framework in [9], which prefetches and pushes videos by cloud-based agents based on social relationships of users.

Cloud computing has been well positioned to provide video streaming services, particularly in the wired Internet because of its scalability and capability [6]. For example, the quality-assured bandwidth autoscaling for video-on-demand streaming based on cloud computing is proposed [8]. However, extending the cloud computing-based services to mobile environments requires more considerations on many factors, e.g., wireless link dynamics, user mobility, and the limited capability of mobile devices [7]. New designs for mobile service on top of mobile cloud computing environments are proposed to virtualize private agents that are in charge of satisfying the requirements (e.g., QoS) of individual users such as Cloudlets [11] and Stratus [12]. The AMES-Cloud framework in [9] proposes a new method of prefetching videos by cloud-based agents and pushing to mobile users cognitively. Thus, we are motivated to design the PreFeed framework by using virtual agents in the cloud to provide prefetching-based RSS services and cognitive delivery for mobile users.

III. PreFeed Framework

A. Introduction of PreFeed Framework

First, we define the terms that are used in this paper. An RSS source is the RSS website with the original content; an RSS feed is a set of many RSS updates or, for example, new articles. In the RSS, there are multiple abstracted information of the news or articles from the RSS source; a particular piece of news or article is called a **content**, which may be a mixture of text, image, audio, and video.

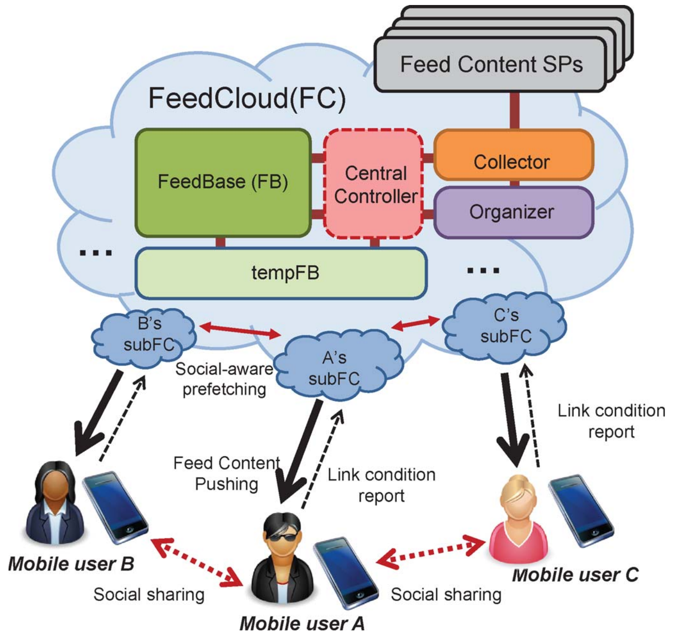


Fig. 1. Illustration of the PreFeed framework.

As illustrated in Fig. 1, PreFeed has one important centralized cloud called **feed cloud (FC)**, which is the key role of feed prefetching and pushing. In the FC, there is the main **feed base (FB)**, which stores all prefetched feed contents, including the XML-based updates of the feed and all text, image, and multimedia contents of the original website. There is also a tempFB, which temporarily stores new feeds, and once it is accessed by a required number of users, it will be moved to FB. There is the **organizer** to make a summary of all RSSs of all users by filtering out duplicated ones, and then the **collector** will fetch the RSS updates and the original content from the feed content SPs. In particular, for each active subscribing user, the cloud virtualizes a smart agent called **subFC** to monitor the user RSS requirement, as well as the wireless link quality, and then to decide how to request the RSS content and how to push content to the mobile user. Due to the elastic computation of dynamic resource allocation of cloud computing, FCs with subFCs are working with optimal performance adaptively to the user demands.

Regarding RSS content pushing, because mobile users have different mobility pattern and link conditions, the subFC will receive the periodical updates of wireless link condition from the users and selectively push content to users adapting to the link condition. Furthermore, due to the social sharing activities in RSS feeds, e.g., Google Reader, one user's friends can mark stars for particular RSS feeds, which informs the user may have a high probability to access the content so that the content will be further fetched for delivery.

B. Cloud-Assisted Prefetching

1) RSS Filtering and Shrinking: In PreFeed, users first share their subscribed RSS lists to the cloud agent subFC, which will report to FC's organizer to shrink the duplicated RSSs (due to the disparity of the RSS popularity) with the requests from other users. Due to the disparity of RSS popularity, it is expected that many duplicated subscriptions will be filtered out and the redundant bandwidth and storage consumption will be maximally reduced. Each subFC will evaluate the update frequency of each RSS feed of each user and then carry out prefetching feed content periodically, which will be discussed later.

2) HTML Parsing and Content Fetching: The RSS feed is only an XML-based file containing the URL address of the RSS site, and if we take HTML request to that address, a list of up-to-date news in format of abstract will be returned. The collector will not only fetch the abstracts of the RSS updates but also **parse** the HTML source code of the original website of the specified document in order to obtain the texts, images, and even embedded videos. How to parse HTML source code is introduced in [23].

Once all the files of the document are obtained, they will be stored in tempFC by extending new names with suffixes while the controller will update the file database FB. The naming system can be referred to the Named Data Networking architecture [24]. Anytime any new request arrives from subFC, the FC will check the tempFB and FB to check if the needed file is already cached by maximal prefix matching. If so, it will be directly reused; otherwise, the collector will fetch from the original website.

C. Cognitive Pushing by Monitoring Link Quality

Every user has her own habit on mobile reading while she may be also in varying mobility conditions. Furthermore, the cellular link condition is fluctuating all the time. So how to find an appropriate time to push the content to the user while many people prefer to not to use cellular data plan but only access to free Wi-Fi for obtaining the RSS contents? When link quality is bad, they should not refresh for new RSS updates due to large amount of energy consumption. Therefore, the mobile client at user's device will periodically report the link condition (whether it is connected to 3G/4G or to Wi-Fi and whether the signal is good or bad). Hence, the cloud agent subFC will check the availability.

Note that when pushing, the RSS contents will be compressed by a zip software development kit and then delivered, as compressing and extracting now consume little CPU resource but may save a lot of transmission bandwidths.

IV. Social RSS Sharing

As studied in [22] and [25], traditional prefetching scheme has low hitting rate as we cannot predict whether the user will access the prefetched content accurately. However, the social relationship is more effective to predict the access probability. By RSS feeds, users can subscribe to well-known famous people and particular content publishers; moreover, there are various types of social activities among users, i.e., **rating** and **sharing** in current social RSS readers, e.g., Flipboard [26].

We define different strength levels for those social activities to indicate the different possibilities that the RSS feed shared by one user may be accessed by the recipients of his/her sharing activities so that subFCs can carry out effective prefetching and push to users local storage. The amount of prefetched segments is mainly determined by the strength of the activities and users' link status.

We classify the activities in RSS services into three kinds regarding the impact of the activities and the potential reacting priority from the point of view of the recipients [27], as denoted by $FeedLv$.

1) **Normal RSS subscription**: This is the basic behavior of RSS services; a user can subscribe to a particular RSS feed collection of some other users based on interests. This interest-driven connectivity between the subscriber and the publisher is considered as "weak" impact as the subscriber may not always watch all updates.
2) **Rating**: A user can always rate stars on an RSS after reading, and the user's friends can obtain this information. In most cases, the rating of friends is important for the user's access. We consider this as "normal" impact.
3) **Direct sharing**: For RSS feeds, a user can directly recommend an RSS feed to particular friend(s) by notice or by mail. The recipients of the message may access the feed with very high probability. This is considered as "strong" impact.

Different strengths of the activities indicate different levels of probability that an RSS will be soon accessed by the recipient. Correspondingly, we also define three prefetching priority levels $PushLv$ regarding the social activities of users on the RSS feeds.

1) "**High**": The RSS update shared by direct sharing will be accessed with a very high probability; hence, we propose to prefetch and push all content documents.
2) "**Mid**": Based on the rating of friends, a user can have a bit higher probability to access the RSS feed; hence, we propose to prefetch and push a part of the feed content, e.g., the full text and images.
3) "**Low**": Because the RSS feeds that are published by subscriptions may be accessed by the subscribers with a not so high probability, we propose to only prefetch and push the text of the RSS content documents.

Prefetching happens among subFC and the FB and, more importantly, will be performed from the subFC in the cloud to the device locally depending on the link quality. If a mobile user is covered by Wi-Fi access, due to Wi-Fi's capable link and low price (mostly for free), subFC can push as much as possible in most cases regarding the prefetching levels. However, if he/she is with a 3G/4G connection, we propose to downgrade the prefetching level to save energy and money.

For example, as shown in Fig. 2, when user A gets an interesting RSS update of a music video (MV) from his/her RSS subscription, prefetching level "mid" should be chosen; however, because A is connected by 3G, prefetching will be downgraded to "low." User B gets direct recommendation of the RSS from A, and thus, B's subVB will prefetch the video at the level of "all." However, B is also connected by 3G; hence, only "mid" level prefetching is triggered. User C sees B's rating activity of the RSS feed while C is connected by Wi-Fi, and thus, C's subVB will push a small part of the content to the device at "low" level.

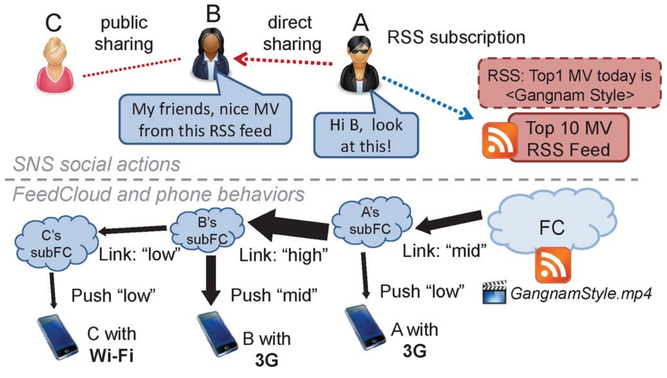


Fig. 2. Illustration of social sharing.

V. Optimization of PreFeed Framework

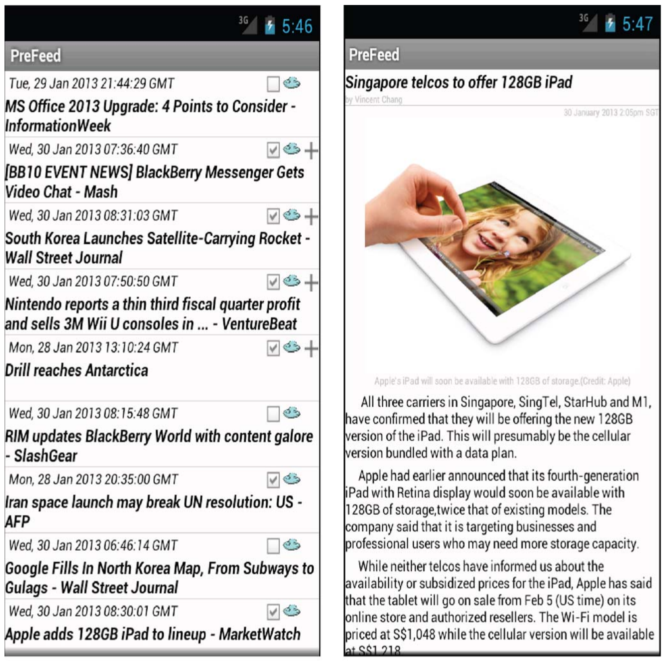
A. Prefetching Frequency

Assume there are totally N users and M RSS feeds. For any user, u_i, $i = [1 \ldots N]$, if he/she subscribes the RSS f_j, $j = [1 \ldots M]$, it will have an average update interval t_{ij}. There are two duty cycles of the subFC: 1) the interval for monitoring the link condition for u_i, which is set to $t_m = \min(t_{ij})$, $\forall j = [1 \ldots M]$; and 2) the interval for the subFC to check a particular RSS feed f_j, which is set to $t_r = \min(t_{ij})$, $\forall i = [1 \ldots N]$. Thus, this means that the users with low update frequency can benefit from caching in PreFeed framework because users with high update frequency have already fetched it.

B. Cognitive Pushing

Algorithm 1 Cognitive Pushing Algorithm

```
repeat
  Sleep for $t_r$
  if $FeedLv == DirectSharing$ then
    $PushLv = High$
    if $FeedLv == Rating$ then
      $PushLv = Mid$
    else
      $PushLv = Low$
    end if
  end if
  Check user link condition from user client's report
  if Link $==$ 3G/4G then
    $DownGrade(PushLv)$
    if Signal $==$ Bad then
      $DownGrade(PushLv)$
    end if
  else
    $//Link == Wi-Fi$
    if Signal $==$ Bad then
      $DownGrade(PushLv)$
    end if
  end if
  Transmit RSS content by $PushLv$
until All RSS feeds in this time slot are transmitted
```



Tue, 29 Jan 2013 21:44:29 GMT
MS Office 2013 Upgrade: 4 Points to Consider - InformationWeek
Wed, 30 Jan 2013 07:36:40 GMT
[BB10 EVENT NEWS] BlackBerry Messenger Gets Video Chat - Mash
Wed, 30 Jan 2013 08:31:03 GMT
South Korea Launches Satellite-Carrying Rocket - Wall Street Journal
Wed, 30 Jan 2013 07:50:50 GMT
Nintendo reports a thin third fiscal quarter profit and sells 3M Wii U consoles in ... - VentureBeat
Mon, 28 Jan 2013 13:10:24 GMT
Drill reaches Antarctica
Wed, 30 Jan 2013 08:15:48 GMT
RIM updates BlackBerry World with content galore - SlashGear
Mon, 28 Jan 2013 20:35:00 GMT
Iran space launch may break UN resolution: US - AFP
Wed, 30 Jan 2013 06:46:14 GMT
Google Fills In North Korea Map, From Subways to Gulags - Wall Street Journal
Wed, 30 Jan 2013 08:30:01 GMT
Apple adds 128GB iPad to lineup - MarketWatch

Singapore telcos to offer 128GB iPad
by Vincent Chang
30 January 2013 2:05pm SGT

Apple's iPad will soon be available with 128GB of storage.(Credit: Apple)

All three carriers in Singapore, SingTel, StarHub and M1, have confirmed that they will be offering the new 128GB version of the iPad. This will presumably be the cellular version bundled with a data plan.

Apple had earlier announced that its fourth-generation iPad with Retina display would soon be available with 128GB of storage,twice that of existing models. The company said that it is targeting businesses and professional users who may need more storage capacity.

While neither telcos have informed us about the availability or subsidized prices for the iPad, Apple has said that the tablet will go on sale from Feb 5 (US time) on its online store and authorized resellers. The Wi-Fi model is priced at S$1,048 while the cellular version will be available at S$1,218

(a) (b)

Fig. 3. Screen captures of PreFeed prototype application. (a) RSS list of PreFeed. (b) News view of PreFeed.

The algorithm of how to cognitively push content to mobile users is shown in Algorithm 1. The basic principle is to offer prefetching and pushing based on $FeedLv$ and $PushLv$, whereas $PushLv$ will be adaptively downgraded due to user's link condition.

VI. Implementation and Evaluation

A. Prototype Implementation and Setting

We evaluate the performance of the PreFeed framework by a prototype implementation in real system. We choose the Korea Telecom U-cloud server (premium account) and utilize the virtual server with six virtual CPU cores (2.66 GHz) and 32-GB memory. In the cloud, we deploy our server application based on Python and Java, including the main core program handling all tasks of the whole FC while it dynamically initializes, maintains, and terminates instances of another subFC agent for each active user. The mobile data service is offered by LG LTE network, which is expected to be able to offer 54-Mb/s bandwidth per user theoretically with about access latency about 20–50 ms. We implement the PreFeed reader client into a mobile platform, Samsung Galaxy SIII, with Android system version 4.0.1. For single test, the RSS sources are collected from Google Reader with 125 popular feeds. Furthermore, for realistic tests, we take a group of 78 students to install our Android application, and they will subscribe to any RSS feed of their interest.

As shown in Fig. 3, the Android-based PreFeed reader application has a page showing all obtained RSS contents and another page to detail the selected RSS feed. The cloud symbol means that the content is directly fetched from the cloud not from remote RSS content SPs. In addition, the checklist icon

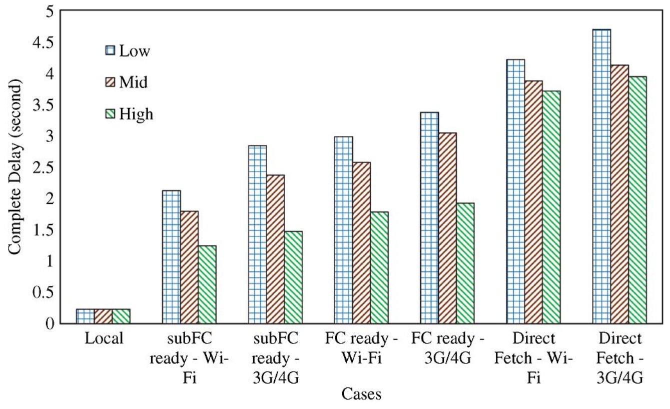


Fig. 4. Complete delays for various cases.

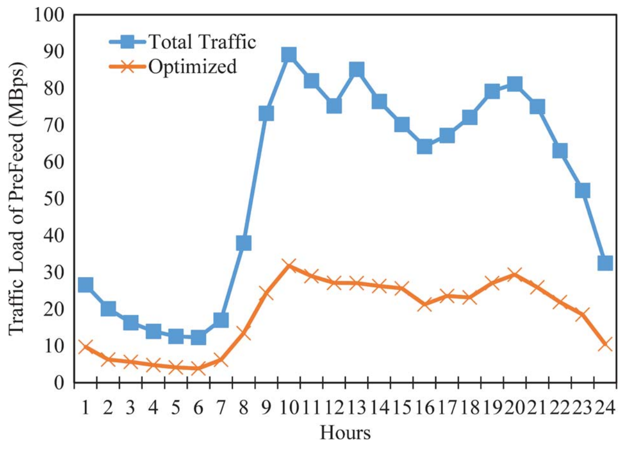


Fig. 5. Traffic load without and with PreFeed.

indicates that the RSS has been prepared by the subFC and thus been pushed to the mobile device before user's access.

B. Performance Measurement and Evaluation

The first test is to verify the system design for a single user by focusing on the complete delay, which is measured as the transmission duration of prefetching and pushing plus the time after the user click to access until the moment that the rest of the whole original document of the RSS feed is downloaded under different $PushLv$ and link conditions. The values are averaged of all users, and this metric is to evaluate the total delay for PreFeed to provide RSS service.

By different settings of $PushLv$ and link conditions, the results, as shown in Fig. 4, indicate that the PreFeed can significantly improve the download speed if the RSS feeds have been requested by other users and thus cached in the FC. Even if we directly request the whole document from the remote SP servers, the complete time is still not a big issue. Because PreFeed can find proper link condition to push to users, the total delay is reduced.

Then, we test how the system performs when there are a group of 78 mobile users. They access RSS feeds during their daily life, and we evaluate the real traffic via the PreFeed service per day. As shown in Fig. 5, the total traffic is the estimated full traffic by assuming if there is no caching, which shows traditional RSS services for multiple users, and the optimized traffic is the realistic traffic going through PreFeed. Obviously, due to the reutilization of RSS content files cached in subFB/FB, a large portion of the traffic, around 43%–74%, is reduced. This popularity disparity effect due to the power law [4] makes significant impact. The 78 students mostly follow several popular RSS feeds; hence, PreFeed reduces the duplicated request for optimization.

VII. Conclusion

In this paper, we have utilized the cloud computing technology to propose a new framework to improve the quality of RSS reading service for mobile users called PreFeed. PreFeed consists of two parts, i.e., cloud-assisted prefetching, which will proactively fetch the multimedia content of the RSS feeds for all subscribed mobile users, and cognitive pushing, which will push the content to mobile users at an appropriate time by evaluating the link quality and user QoS requirements. Furthermore, the social impact among users is considered. We implement a prototype of the PreFeed framework to evaluate its performance. It is shown that cloud computing can effectively facilitate feed prefetching and cognitive pushing for mobile users; a large portion of traffic load (around 43%–74%) due to redundant downloads can be reduced. In the future, we will try to improve and optimize the PreFeed framework for large-scale deployment.

References

[1] X. Wang, A. V. Vasilakos, M. Chen, Y. Liu, and T. T. Kwon, “A survey of green mobile networks: Opportunities and challenges,” *ACM/Springer MONET*, vol. 17, no. 1, pp. 4–20, Feb. 2012.

[2] M. Chen, “MM-QoS for BAN: Multi-Level MAC-Layer QoS Design in Body Area Networks,” in *Proc. IEEE Globecom*, Atlanta, GA, USA, Dec. 9–13, 2013.

[3] RSS 2.0 Specification, Advisory R. S. S. Board, Jun. 2007.

[4] Power Law. [Online]. Available: http://en.wikipedia.org/wiki/Power_law

[5] X. Li, J. Yan, Z. Deng, L. Ji, W. Fan, B. Zhang, and Z. Chen, “A Novel Clustering-based RSS Aggregator,” in *Proc. 16th Int. Conf. World Wide Web*, 2007, pp. 1309–1310.

[6] M. Chen, Y. Wen, H. Jin, and V. Leung, “Enabling technologies for future data center networking: A primer,” *IEEE Netw.*, vol. 27, no. 4, pp. 8–15, Jul. 2013.

[7] C. Lai, H. Chao, Y. Lai, and J. Wan, “Cloud-assisted real-time transrating for http live streaming,” *IEEE Wireless Commun.*, vol. 20, no. 3, pp. 62–70, Jun. 2013.

[8] D. Niu, H. Xu, B. Li, and S. Zhao, “Quality-assured cloud bandwidth auto-scaling for video-on-demand applications,” in *Proc. IEEE INFOCOM*, 2012, pp. 460–468.

[9] X. Wang, M. Chen, T. T. Kwon, L. T. Yang, and V. C. M. Leung, “AMES-Cloud: A framework of adaptive mobile video streaming and efficient social video sharing in the clouds,” *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 811–820, Feb. 2013.

[10] L. Zhou and H. Wang, “Toward blind scheduling in mobile media cloud: Fairness, simplicity, and asymptotic optimality,” *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 735–746, Jun. 2013.

[11] N. Davies, “The case for VM-based cloudlets in mobile computing,” in *IEEE Pervasive Comput.*, Oct.–Dec. 2009, vol. 8, no. 4, pp. 14–23.

[12] B. Aggarwal, N. Spring, and A. Schulman, “Stratus: Energy-efficient mobile communication using cloud support,” in *Proc. ACM SIGCOMM*, 2010, pp. 477–478.

[13] C. Ji and J. Zhou, “A study on recommendation features for an RSS reader,” in *Proc. Int. Conf. CyberC*, Oct. 2010, pp. 193–198.

[14] J. J. Samper, P. A. Castillo, L. Araujo, J. J. Merelo, Cordón Ó., and F. Tricas, “NectaRSS, an intelligent RSS feed reader,” *J. Netw. Comput. Appl.*, vol. 31, no. 4, pp. 793–806, Nov. 2008.

[15] A. Das, M. Datar, and A. Garg, “Google news personalization: Scalable online collaborative filtering,” in *Proc. 16th Int. Conf. World Wide Web*, 2007, pp. 271–280.

[16] Y. Kim, J. W. Lee, S. R. Park, and B. C. Choi, “Mobile advertisement system using data push scheduling based on user preference,” in *Wireless Telecommun. Symp.*, 2009, pp. 1–5.

[17] G. Cugola and H.-A. Jacobsen, “Using publish/subscribe middleware for mobile systems,” *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 6, no. 4, pp. 25–33, Oct. 2002.

[18] S. Zhao, P. P. C. Lee, J. C. S. Lui, X. Guan, X. Ma, and J. Tao, “Cloud-based push-styled mobile botnets: A case study of exploiting the cloud to device messaging service,” in *Proc. ACSAC*, Dec. 2012, pp. 119–128.

[19] G. Yunwen, W. Shaochun, Y. Bowen, and L. Jiazheng, “The methods of data prefetching based on user model in cloud computing,” in *Proc. Int. Conf. iThings/CPSCom*, Oct. 2011.

[20] B. Cheluvaraju, A. S. R. Kousik, and S. Rao, “Anticipatory retrieval and caching of data for mobile devices in variable-bandwidth environments,” in *Proc. IEEE Int. SysCon*, Apr. 2011, pp. 531–537.

[21] B. Li and X. Jia, “Caching and prefetching for web content distribution,” in *Proc. Comput. Sci. Eng.*, Jul./Aug. 2004, vol. 6, no. 4, pp. 54–59.

[22] B. D. Higgins, J. Flinn, T. J. Giuli, B. Noble, C. Peplin, and D. Watson, “Informed mobile prefetching,” in *Proc. ACM MobiSys*, 2012, pp. 1–14.

[23] Y. Yang and H. Zhang, “HTML page analysis based on visual cues,” in *Proc. 6th Int. Conf. Document Anal. Recog.*, 2001, p. 859.

[24] B. Han, X. Wang, N. Choi, T. T. Kwon, and Y. Choi, “AMVS-NDN: Adaptive mobile video streaming and sharing in wireless named data networking,” in *Proc. INFOCOM NOMEN Workshop*, 2013, pp. 1–6.

[25] X. Wang, T. T. Kwon, Y. Choi, H. Wang, and J. Liu, “Cloud-assisted adaptive video streaming and social-aware video prefetching for mobile users,” *IEEE Wireless Commun. Mag.*, vol. 20, no. 3, pp. 72–79, Jun. 2013.

[26] Flipboard. [Online]. Available: http://www.lipboard.com/

[27] Z. Wang, L. Sun, C. Wu, and S. Yang, “Guiding internet-scale video service deployment using microblog-based prediction,” in *Proc. IEEE INFOCOM*, 2012, pp. 2901–2905.

Xiaofei Wang received the B.S. degree from Huazhong University of Science and Technology, Wuhan, China, in 2005 and the M.S. and Ph.D. degrees from Seoul National University, Seoul, Korea, in 2008 and 2013, respectively.

His current research interests are social-aware multimedia services in cloud computing, cooperative backhaul caching, and traffic offloading in mobile content-centric networks.

Dr. Wang was a recipient of the Korean Government Scholarship for Excellent Foreign Students in the IT field by the National IT Industry Promotion Agency from 2008 to 2011 and the Global Outstanding Chinese Ph.D. Student Award in 2012.

Min Chen (SM'09) is currently a Professor with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China. He was an Assistant Professor with the School of Computer Science and Engineering, Seoul National University (SNU), Seoul, Korea, from September 2009 to February 2012. He was a Postdoctoral Fellow with the Department of Electrical and Computer Engineering, The University of British Columbia (UBC), Vancouver, BC, Canada, for three years. Before joining UBC, he was a Postdoctoral Fellow with SNU for one and half years. He has more than 170 paper publications.

Prof. Chen is a Technical Program Committee Member for the IEEE Conference on Computer Communications (INFOCOM) 2014. He is a Guest Editor for *IEEE Network*, *IEEE Wireless Communications*, etc. He is the Symposium Cochair of the IEEE International Conference on Communications (ICC) 2012 and the IEEE ICC 2013. He is the General Cochair of the IEEE International Conference on Computer and Information Technology 2012. He is a Keynote Speaker for CyberC 2012 and Mobiquitous 2012. He was a recipient of the Best Paper Award from the IEEE ICC 2012 and the Best Paper Runner-up Award from QShine 2008.