ICOW: Internet Access in Public Transit Systems

Se Gi Hong *, SungHoon Seo *, Henning Schulzrinne *, and Prabhakar Chitrapu †
*Columbia University, New York, NY
{segihong, hoon, hgs}@cs.columbia.edu
†InterDigital Communications, LLC, King of Prussia, PA
Prabhakar.Chitrapu@InterDigital.com

***Abstract*—When public transportation stations have access points to provide Internet access to passengers, public transportation becomes a more attractive travel and commute option. However, the Internet connectivity is intermittent because passengers can access the Internet only when a bus or a train is within the networking coverage of an access point at a stop. To efficiently handle this intermittent network for the public transit system, we propose Internet Cache on Wheels (ICOW), a system that provides a low-cost way for bus and train operators to offer access to Internet content. Each bus or train car is equipped with a smart cache that serves popular content to passengers. The cache updates its content based on passenger requests when it is within range of Internet access points placed at bus stops, train stations or depots. We have developed a system architecture and built a prototype of the ICOW system. Our evaluation and analysis show that ICOW is significantly more efficient than having passengers contact Internet access points individually and ensures continuous availability of content throughout the journey.**

I. Introduction

Providing Internet access to passengers makes public transit more attractive as an alternative to commuting or traveling by car. Thus, many public transit organizations are trying to provide such access. There are several options:

1 They can install wireless routers on transit vehicles that combine a 3G or 4G cellular data interface with local IEEE 802.11 connectivity. This option is becoming common for long-distance travel coaches. However, transit operators or passengers have to pay for the cellular data service and the bandwidth per passenger is low.

2 In the future, 4G networks may become sufficiently ubiquitous and cheap so that individual passengers can use their own devices, such as smartphones or tablets, but it is unlikely that Internet access in subways will occur before 2020 in many countries.

3 They can install WLAN access points at bus stops and train stations, allowing connectivity for passengers. However, since such connectivity may only last for a short time within limited network coverage of an access point, passengers would experience only an intermittently connected network service. Despite this intermittent network service, the free service encourages passengers to use this WLAN network.

We focuses on the last option, developing a low-cost way for public transit systems to access web applications. When passengers carry a hand-held mobile device which is equipped with a WLAN interface, they can freely use Internet service without activating a cellular data plan. This option is suitable for public transit systems in dense urban area where the distance between stops is short. This option can, in particular, be used in subway systems where there is no cellular network connectivity. Even though in some metropolitan cities in Asia, such as Seoul and Tokyo, network service providers provide cellular connection in subway, some cities in America and Europe, such as New York city and London, do not offer cellular connectivity.

These intermittently connected networks suffer from resource limitations, especially network connection period and bandwidth limitation. When requests for content are not handled during a network connection period because the stopping time is not long enough, the non-handled requests need to be queued until the next stop. This increases content retrieval delay. In addition, there is a flash crowd problem at the beginning of network connection period because passengers simultaneously try to access the network when they get Internet connectivity. These simultaneous attempts cause network contention problems that degrade link throughput.

We propose Internet Cache on Wheels (ICOW) for Internet content in public transit systems. ICOW is a cache-based system that can be deployed on a transit vehicle. This cache avoids retrieving the same content twice from the Internet during a network connection period. Hence, it reduces network resource usage. Secondly, by placing this smart cache on a vehicle and allowing it to work as a proxy node, this system resolves the network contention problem.

ICOW is not only a simple caching system, it also provides functions to increase bandwidth availability, increase system efficiency, and decrease content delivery delay. The core features that the ICOW system provides are as follows:

- **Increase bandwidth availability.** The ICOW system provides a channel division mechanism that eliminates single channel interference and a request aggregation mechanism that reduces bandwidth usage.
- **Increase system efficiency.** The ICOW system queues requests. This queuing mechanism allows passengers to send requests even in the absence of an Internet connection. ICOW also provides related cached web content to passengers when their requested content has not yet been cached. The asynchronous notification of content availability in the ICOW system makes the system easier to use in the sense that passengers do not need to check repeatedly whether the requested content is available.

- **Decrease content delivery delay.** The ICOW system schedules requested content retrieval. When the network connecting time is not long enough to handle all requests, some requests might be handled during the next connecting period. To decrease this content delivery delay, ICOW applies popularity-based scheduling, and it also prefetches related content.

We describe our design of the system architecture and implementation of the system. Our performance evaluation shows that ICOW obtains more content in a given time than a direct access system (where each passenger directly connects to an AP) when the number of passengers is above a certain number. We analytically show that our system supports the distribution of network resource use in time domain and reduces the bandwidth usage during network connection periods. Our simulation results show that our popularity-based scheduling algorithm reduces average content delivery delay while simplifying implementation.

The remainder of this paper is organized as follows. In Section II, we define a public transportation model that we use. In Section III, we describe some of the issues that need to be addressed in this public transportation model. We discuss related work in Section IV. We provide our approaches to resolve the problems in Section V. In Section VI, we show architecture and implementation details. Finally, we evaluate the system performance in Section VII.

II. Public transit network access model

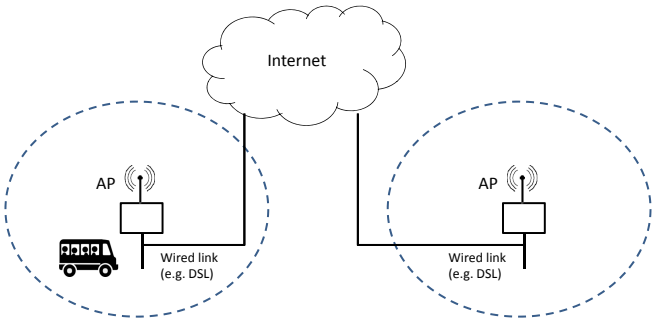


Fig. 1. Public transit system. The dotted line presents the network coverage of an AP.

We define a public transit model for intermittent network access where dwell time (standing time), running time and passenger travel time are the major parameters. Fig. 1 shows the public transit model.

A. Stops and stations

We assume that every bus stop, subway stop, or train station is equipped with an access point (AP). These APs provide Internet access service to passengers while the bus or train is within the vicinity of the AP. When passengers are within the network coverage of an AP, it is called a network connection period. Otherwise, they are in a network disconnection period.

B. Passenger

We assume that passengers carry handheld mobile devices or laptop computers which contain an IEEE 802.11 wireless network interface card (NIC). They greedily access the Internet such as web applications while a bus or train is within the vicinity of the AP at stops. They can find whether they are in the network coverage of an AP either by checking signal strength or by using other network scanning tools to find an AP. The bandwidth between passengers and an AP is limited, and the amount of data that passengers can upload and download through an AP depends on the vehicle stopping period. The passengers' travel time is limited, so they get off after passing some random numbers of stops.

C. Vehicle

In our discussion, we refer to vehicles as any kind of public transit vehicles, such as a bus, subway, light rail, ferry, or train. Vehicles make frequent stops to pick up and discharge passengers.

III. Problems

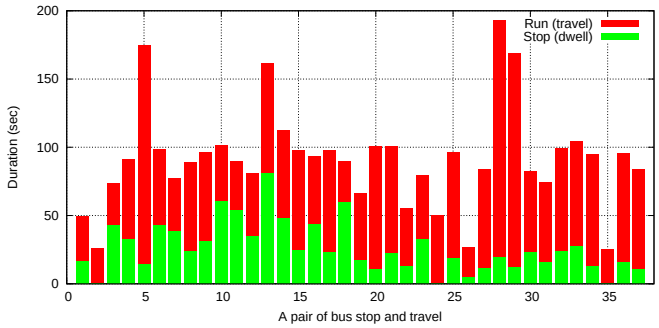
In this section, we describe the problems that occur in public transit network access. Since passengers can access web content when a transit vehicle is within the vicinity of access points, they would experience interrupted Internet connection. Thus, this public transit network is an intermittently connected network.

A. Network resource limitation

We measured bus stop and travel time in New York City with the results shown in Fig. 2(b). We consider New York City typical of a dense metropolitan area. The result indicates that the average bus dwell time is much lower than the average bus travel time: the average bus dwell time is 26 seconds while the average bus travel time between two stops is 65.4 seconds. Because a network connection is established when a bus is approaching a bus stop and after it starts moving, we measured the average network connection period by setting up a laptop computer with an external antenna at a bus stop and carrying another laptop computer on a bus. We assume that every bus stop has an AP. The measurement result indicates that the network connection period is 16 seconds larger than the bus dwell time. Hence, the average network connection period is about 42 seconds while the average network disconnection period is about 49.4 seconds.

This measurement raises an important problem, limited network resources. Passengers can use network connection only when vehicles are within the network coverage of an AP at each stop. Because of this limited network connection time, some of passengers would experience that they cannot use the Internet when many other passengers use the Internet, so that they should wait until the next stop (in this measurement, they wait 49.4 seconds) to use the Internet. This interrupted connectivity increases the delay to use web applications such as reading online newspapers.

(a) MTA M104 bus route, Manhattan, New York City. Blue circles are bus stops.



(b) Bus stop and travel time

Fig. 2. Bus stop and travel time measured in Manhattan, New York City in November 2009. Average bus dwell time is 26 sec and average bus travel time between stops is 65.4 sec.

B. Network throughput degradation

At the beginning of a network connection period, passengers try to simultaneously establish a link-layer connection to an AP at a stop and use web applications. This simultaneous connections cause network contention and result in degrading the overall network link throughput. It is worth noting that the network throughput, especially in the contention-based medium access technique like IEEE 802.11 WLAN, is significantly affected by the number of contending nodes. The authors of [1][2] analyzed the throughput in the IEEE 802.11 Distributed Coordination Function (DCF) mechanism and showed that the throughput degrades as the number of contending nodes increases. When one node occupies radio resource to transmit its frame, other contending nodes should wait for the next transmission opportunity to send their frame. The next transmission opportunity is randomly chosen between the range of 0 and current contention window size which is exponentially increased whenever the nodes fail to win the medium in a transmission opportunity. Upon contending to transmit a frame, the contention window size (by default, the initial contention size is 31) exponentially increases until it reaches the maximum size 1023. Therefore, the average backoff period increases as contention increases. As a result, link throughput degrades by about 25 % as the number of contending node increases from 1 to 40 when each node has the same transmission opportunity.

To confirm these results, we simulated the impact of contention on throughput using the OPNET simulator [3]. Fig. 3 depicts the throughput variation over a 40 second simulation time as the number of contending nodes (N) changes across the set, $N = \{1, 10, 20, 30, 40\}$. In this simulation, we fixed the channel data rate (CR) as 11 Mb/s and make each N nodes individually start to download a chunk of content of size $S = 30$ Mbytes from a server through a single TCP session using the FTP `GET` command. Simulation result shows that the wireless link provides less throughput as the number of users who compete for the bandwidth increases. Comparing to the result of $N = 1$, the throughput degrades about 11 % and 25 % for $N = 20$ and $N = 40$, respectively, when the link is saturated (after 10 seconds). This simulation result shows that the throughput degradation is not negligible, especially when network bandwidth is limited.

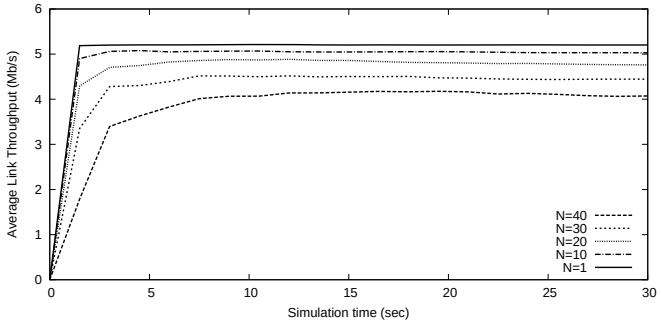


Fig. 3. The effect of contention on throughput of N FTP `GET` flows. Channel rate is 11 Mb/s. Each requested content is 30 Mbytes.

IV. Related work

Cache-based systems. 7DS [4], the earlier project developed, aims to provide systems for web access to retrieve information and message delivery in intermittently connected networks. This system allows mobile devices to exchange cached information and deliver messages in a highly mobile environment. Even though there is no Internet connectivity, mobile users can retrieve content, which other mobile users have already cached using the web-based user interface of the 7DS system. However, it does not address the resource limitation problem and network degradation problem that occur between a mobile device and an AP.

S. Pack et al. [5] describe cache data access algorithms introducing a proxy cache system. In this system, a wireless router is installed on transit vehicles that combine a cellular data interface with local IEEE 802.11 connectivity. The wireless router has a proxy cache to enhance data access. They propose new algorithms of cache replacement and cache consistency in order to improve cache hit ratio and reduce latency in obtaining data. However, this work does not handle intermittently connected networks, so it suffers from the lack of a system that optimizes the usage of limited resources between the proxy cache and Internet AP in intermittently connected networks.

Proxy-based systems. A. Balasubramanian et al. [6] propose Thedu to enhance interactive web search in intermittent mobile-to-Internet AP connectivity environments. They use a server-side proxy (Internet proxy) that prefetches related web pages for web searching requests to reduce the web search delay in intermittently connected networks. However, this system has two problems because the server-side proxy that they use is on the Internet side. It cannot resolve the problem of network throughput degradation, and users cannot use the server-side proxy when there is no Internet connectivity.

J. Ott et al. [7] present a HTTP-over-DTN mechanism using bundles. Getting one web page requires multiple requests and responses because a web page contains many objects, so HTTP does not properly work in intermittently connected networks. To mitigate this problem, this system uses bundles where multiple responses containing all the objects of a web page are sent in a bundle for a single request. Even though this bundling provides an enhanced performance in terms of the delay for getting the amount of web pages given a limited connection time, it does not resolve the network throughput degradation problem because this system uses a server-side proxy. Hence, it also has the same problem as Thedu for our applications.

Routing and scheduling. Since there is a constraint on bandwidth in intermittently connected networks, data scheduling algorithms have been proposed to select which data will be delivered first when mobile devices encounter other peers. These algorithms are based on mobility patterns of peer nodes [8][9], social structures (such as popularity and community) [10][11], and/or personal relationship [12]. However, these algorithms focus on how to forward or replicate data to decrease the delivery delay and increase the success rate to deliver them to specific destinations. Hence, they do not address how to fetch content efficiently from the Internet in intermittently connected networks.

V. System design overview

In this section, we present an overview of the ICOW system. The main objective of ICOW is to increase content availability, system usability and satisfaction of passengers by deploying ICOW nodes on public transit vehicles. This ICOW system is used to access web applications and content. Fig. 4 presents

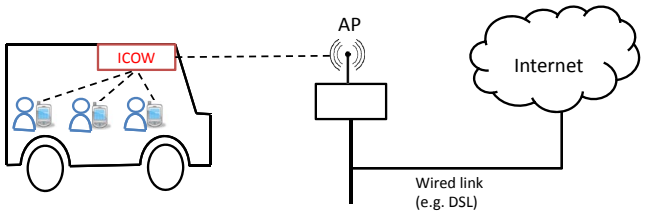


Fig. 4. ICOW system. A vehicle has an ICOW node. Dotted lines present wireless network connection and a solid line presents a wired network connection.

the ICOW system. Passengers access web applications through an ICOW node on a vehicle. An AP at stops provides Internet connection to the ICOW system.

A. Caching proxy-based mechanism

ICOW nodes provide a caching mechanism to reduce the bandwidth usage by eliminating retrieval of duplicate web content from the Internet during a network connection period. Indeed, as discussed in Section III-A, this intermittently connected network in public transit systems suffers from limited bandwidth. However, by using this caching mechanism, the ICOW system alleviates the limited bandwidth problem.

In addition, the ICOW node works as a web proxy server. It establishes the Internet connection between passengers and an AP. Section III-B discussed that network throughput decreases as the number of competing nodes increases. When there are a large number of passengers who want to access the Internet, the network throughput degrades. However, using a proxy mechanism, the ICOW system can resolve the throughput degradation problem by reducing the number of competing nodes to one.

B. Channel division

As shown in Fig. 4, there are two wireless links: the link between passengers and an ICOW node, and the link between an ICOW node and an AP. In wireless networks, when one pair of a sender and a receiver occupies a wireless medium, others have to wait until the link is idle. Hence, if the two wireless links of the ICOW system have the same channel, two simultaneous transmissions in the two links cannot be allowed. This single channel mechanism degrades the overall throughput of the two links.

To overcome this single channel problem, ICOW uses a channel division mechanism as described in Fig. 5. ICOW nodes have two network interface cards, operating on different channels. One of them is used to communicate with passengers and the other communicates with an AP. Hence, there is no single channel interference between the two wireless links, so the overall throughput of the two links increases.

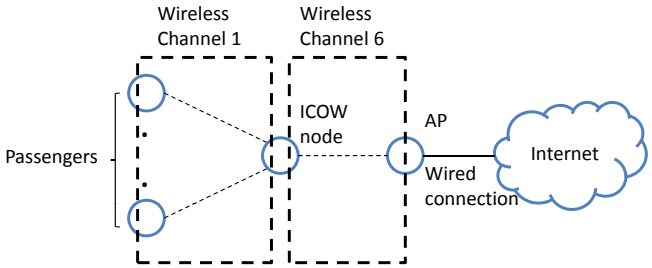


Fig. 5. Channel division.

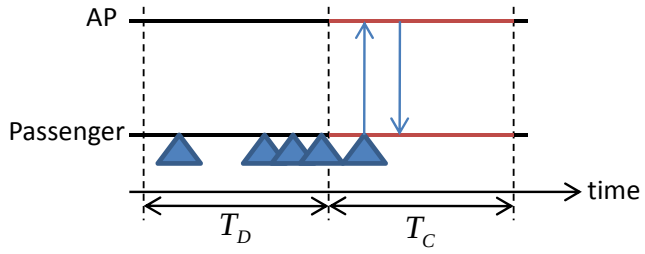
C. Queuing Requests

Existing HTTP proxy servers do not properly work in intermittently connected networks. When there is no Internet connection, a proxy server simply returns a failure, indicating that it has a network connection problem. Hence, the clients should keep retrying until there is an Internet connection. This paradigm causes inconvenience in the sense that clients have to manually polling for the Internet connection every moment before sending requests as shown in Fig. 6(a). If clients are doing other jobs while waiting for an Internet connection, they have to occasionally stop them to check the Internet connectivity and transmit their requests when the Internet connection is alive.

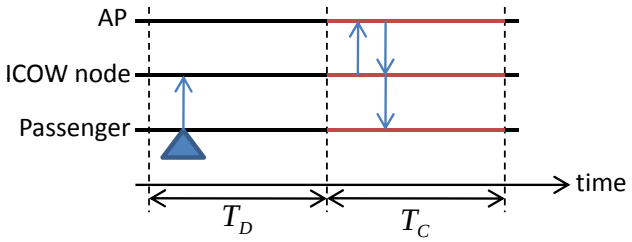
To overcome this inefficiency of the manual system, we propose a queuing system which queues clients' requests until the system fetches the requested content from the Internet. This system allows passengers to send requests even when the ICOW node has no Internet connection. The ICOW node queues the requests and retrieves requested content when it has an Internet connection and distributes the contents to passengers as shown in Fig. 6(b).

In addition, the ICOW system aggregates requests. Existing HTTP proxy servers make connections to a web server for each request. Hence, when multiple passengers request the same content, each connection for each request wastes bandwidth. Especially with limited bandwidth, this duplication of fetching the same content from the Internet significantly degrades performance of the system.

To reduce this waste of bandwidth, the ICOW system has a request aggregation functionality. When an ICOW node receives multiple requests for the same URL during network disconnection period, it aggregates the requests. Thanks to a caching mechanism of an ICOW node, it can retrieve the content from the Internet only once and store the content in a local cache, as shown in Fig. 7.



(a) Manual system.



(b) Queuing system.

Fig. 6. The shift of requesting point in time domain from the perspective of passengers. The triangle represents the actual trying point to transmit a request. The arrow is the actual data transmission. T_D is a network disconnection period and T_C is a network connection period.

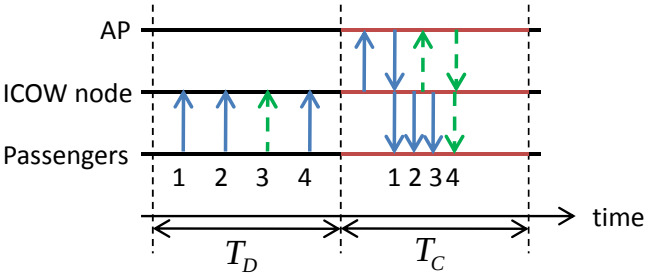


Fig. 7. Request aggregation. i is a passenger i. Solid arrow is for content A and dotted arrow is for content B. T_D is a network disconnection period and T_C is a network connection period.

D. Related content retrieval and asynchronous notification

One of advanced caching mechanisms of the ICOW system is to provide related content to passengers when their requested content is not available (i.e., has not yet cached). For example, when a passenger requests the New York Times web page but it is not cached, the passenger might be interested in other news web sites which have already been cached, say, the CNN web site.

However, passengers might still worry that they may miss their requested content, so they might frequently check whether their content has been cached. To make the system easier to use, the ICOW node notifies passengers when their content has been cached without having to manually check for updates. This asynchronous notification is based on a server-initiated HTTP push mechanism.

Fig. 8 shows how other cached content is provided to

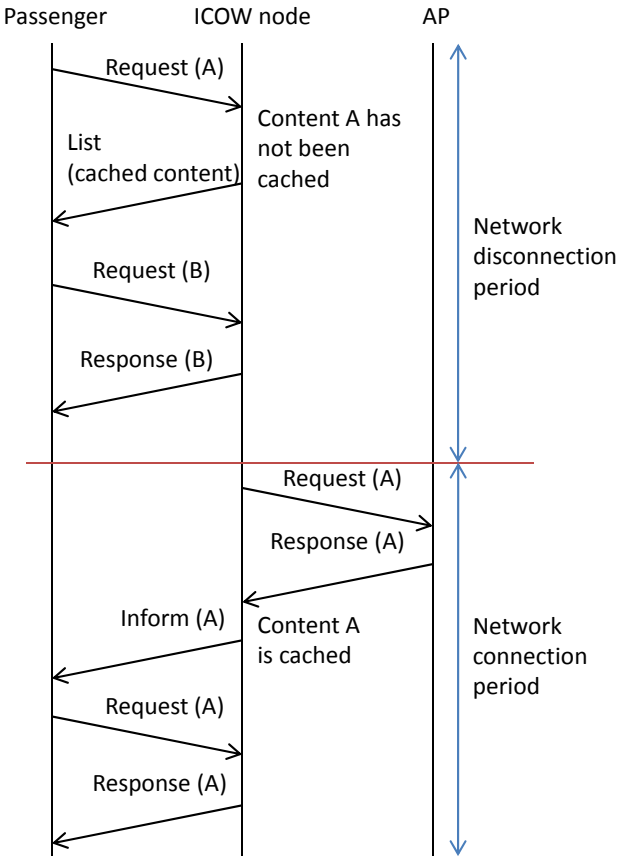


Fig. 8. Related content retrieval and asynchronous notification

passengers and how passengers receive their requested content. A passenger requests content A, but the content has not been cached yet and there is no Internet connection. The ICOW node provides a web page with a list of related cached content. Because the passenger is interested in content B in the list and requests it, the ICOW node provides the cached content. By using asynchronous notification of cached content, when the requested content A is cached (after the ICOW node has connected to the Internet), it pushes the notification to the passenger, indicating that it cached the content. If the passenger is still interested in the content, the passenger simply browses the content.

E. Scheduling algorithm

1) Popularity-based scheduling: When network connecting time is not long enough to handle all the requests, some passengers might experience a long delay in getting requested content or not receive the content before they get off. Hence, we need a scheduling mechanism to decrease content delivery delay. We use popularity-based scheduling in which the request with highest frequency has the highest priority. Because there might be several requests for the same content, popularity-based scheduling efficiently reduces bandwidth usage and thereby reduces average content delivery delay. To break a tie, we apply normalized most-recent-first (MRF) algorithm in which the most recently requested content has the highest priority.

Combining the popularity and MRF algorithms, we can calculate the value of content that is used to determine the priority. The highest value the content has, the highest priority the content has. The value, $V_i(t)$, of content i at time t is

$$V_i(t) \quad = \quad N_i(t) + \frac{\sum_{j=1}^{N_i(t)} T_{ij}}{t \cdot N_i(t)},$$

where $N_i(t)$ is the number of requests for content i at time t and T_{ij} is the requesting time of content i of passenger j ($T_{ij} < t$). Time t can be the sequence number of stops from a vehicle depot or the monotonically increasing time whose reference point is the starting time at a depot.

2) Prefetching related content: Generally, web pages have hyperlinks that point to other web pages, and there is a high likelihood that passengers want to visit those linked pages from their current web pages. For example, after reading headlines of news web pages, passengers may read the articles in more detail, clicking the links of the articles. In public transit systems, however, when passengers want to read other articles after reading the current web page, the vehicle might already leave the stop. Thus, they have to wait until the next network connection period. If the traveling time of a vehicle to the next stop is long, the waiting time to obtain the article becomes high.

To reduce this waiting time, we use a prefetching mechanism. When the system obtains a web page, if there are no more requests from passengers in a queue, it prefetches pages pointed to by hyperlinks (HTML <a> elements) on the received web page. Compared to requests from passengers, this prefetching has a lower priority. Fig. 9 shows priority queuing.

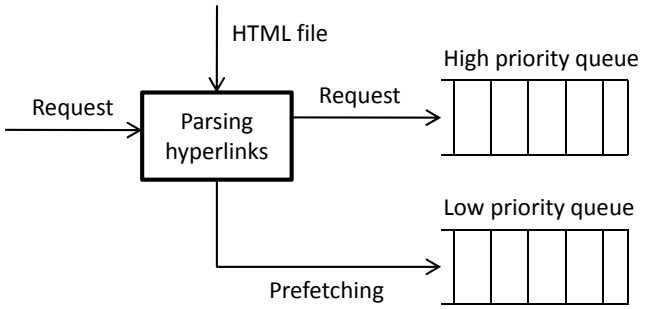


Fig. 9. Priority queuing for requests and prefetching.

Even though prefetching has a low priority, the system needs to reduce the number of prefetching web pages because the number of hyperlinks is significant in most web pages, especially front pages. Hence, we reduce the number of prefetching pages by applying rules: (1) to eliminate advertisements, prefetched pages should be in the same domain as the requested pages and (2) to save bandwidth, prefetched pages should not be media elements (video or audio files).

F. Passenger privacy

Since all requests and responses pass through the ICOW node, there might be a privacy issue. Indeed, passengers do not want their sensitive information to be cached and be released to others. Hence, ICOW needs a mechanism to protect privacy.

To control caching, ICOW uses an explicit cache control mechanism that is provided by HTTP/1.1. HTTP/1.1 introduces the `Cache-Control` header to protect privacy [13]. There are two types of `Cache-Control` directives for privacy, namely `private` response directive and `no-store` request and response directive. The `private` response directive prevents a shared cache, such as a cache in a proxy, from storing responses. However, a user specific, personal cache such as caches in a browser, can cache responses. The `no-store` request and response directive prevents requests and responses from being cached in any caches along a data path. This is useful to prevent any sensitive content from being released to the public. By checking the `Cache-Control` header, ICOW decides whether it caches content or not, so that it protects passengers' privacy.

VI. System Architecture and Implementation

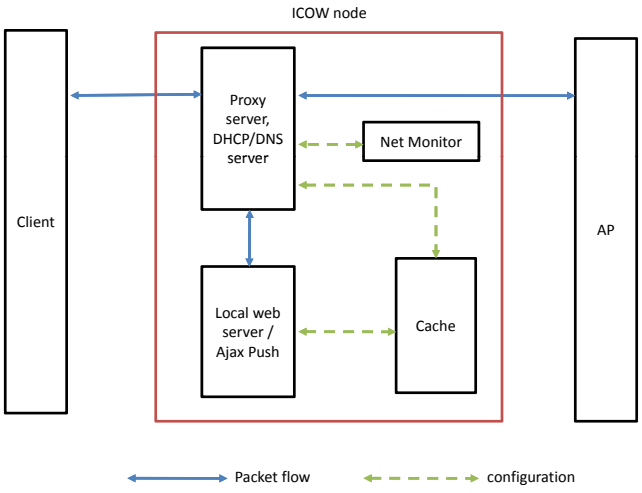


Fig. 10. System architecutre

The system architecture follows from the description above. An ICOW node consists of four software components: network monitor, proxy server, cache and local web server. Fig. 10 shows the system architecture of an ICOW node. Fig. 11 shows a flow chart that illustrates the implementation of the mechanisms described so far. We describe the major components in more detail below.

A. Network monitor

The main role of network monitor is to determine whether the ICOW node can connect to an AP. The network monitor periodically checks whether network connectivity is available and notifies the proxy server component. If connectivity is available, the ICOW node tries to fetch content through the AP.

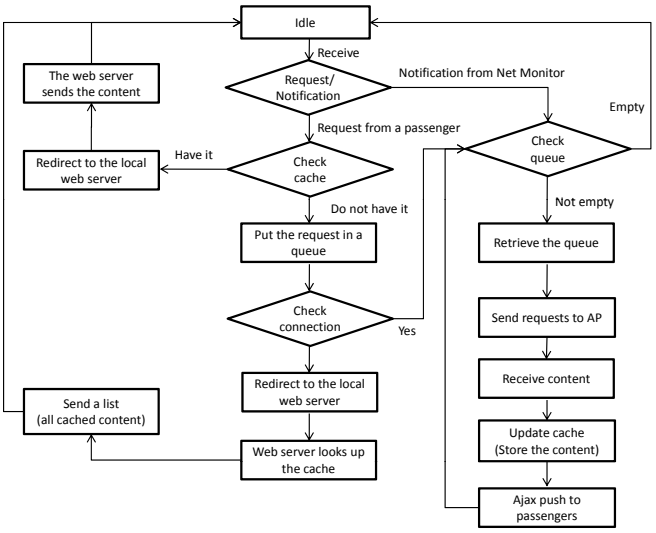


Fig. 11. Flow chart in an ICOW node.

We have implemented the network monitor modules for two platforms, Windows and Linux, because these modules require kernel specific API to check for link connectivity. Windows version of the network monitor module internally calls a driver level function call, `DeviceIoControl`, through the NDIS (Network Driver Interface Specification) and obtains the information of network interface card (NIC). On the other hand, for Linux, we use kernel level system call, `iwgetid`, to check the status of NIC and the IP address that is bound to the MAC address of the NIC.

B. HTTP proxy server

The HTTP proxy server handles all incoming and outgoing packets.

We use the `Muffin` [14], an open source web proxy software. We add four features to `Muffin`: redirecting to local web server, HTTP request scheduling, request queuing, and web caching. When the proxy server receives HTTP requests from passengers, it redirects the requests to the local web server if the local cache has the content. Otherwise, the requests are queued to be handled during network connection periods. Based on the popularity of requests, the priority of the requests is determined, as described in Section V-E1. Proxy server component triggers web caching that downloads requested content in the queue when it has a network connection.

ICOW supports automatic discovery and configuration of the proxy server using Proxy Auto-Config (PAC) [15] and Web Proxy Auto-Discovery Protocol (WPAD) [16]. PAC file[1] is

[1]The PAC file has a JavaScript function, `FindProxyForURL (url, host)` where `url` is the requested URL, and `host` is the hostname extracted from the URL. This function returns `host` and port number of a proxy server. In our system, the return value is the IP address of the ICOW node and port number for the proxy server.

used to configure a proxy server, and WPAD protocol[2] is used to find automatically the location of the PAC file. Using a PAC file and WPAD protocol, passengers can automatically discover and configure a proxy server.

C. Web cache

To eliminate the repetition of the same content retrieval from the Internet, caching is used. It downloads the requested web pages, including HTML files, images, CSS files and JavaScript files. In addition, it prefetches other content pointed to by hyperlinks in the same domain of requested web pages. The list of cached content is accessible to a proxy server and a local web server.

D. Local web server

The local web server provides the web-based user interface to passengers, informing the availability of requested content and providing other cached content when the requested content is not available during a network disconnection period.

The local web server is built on `Apache Tomcat` [17], an open source software of Java Servlet and JavaServer, with `Ajax Push` [18], a server-initiated push mechanism. The push mechanism is for asynchronous notification to inform updated information to passengers. Visiting a guide page, passengers are able to see all the cached content. An HTML `iframe`[3], along with delivered cached pages, is used to show the update in a passenger's browser.

Fig. 12 shows the browsers of passengers. A passenger receives a guide page which includes a list that the ICOW node has cached when there is no Internet connection and the requested content (New York Times) has not yet been cached (Fig. 12(a)). While the passenger is waiting to get requested content, the passenger can read cached pages (CNN web page in Fig. 12). If the ICOW node caches the requested page (New York Times in this figure), the passenger browses the notification in its `iframe` (Fig. 12(c)).

VII. Performance Evaluation

A. Throughput

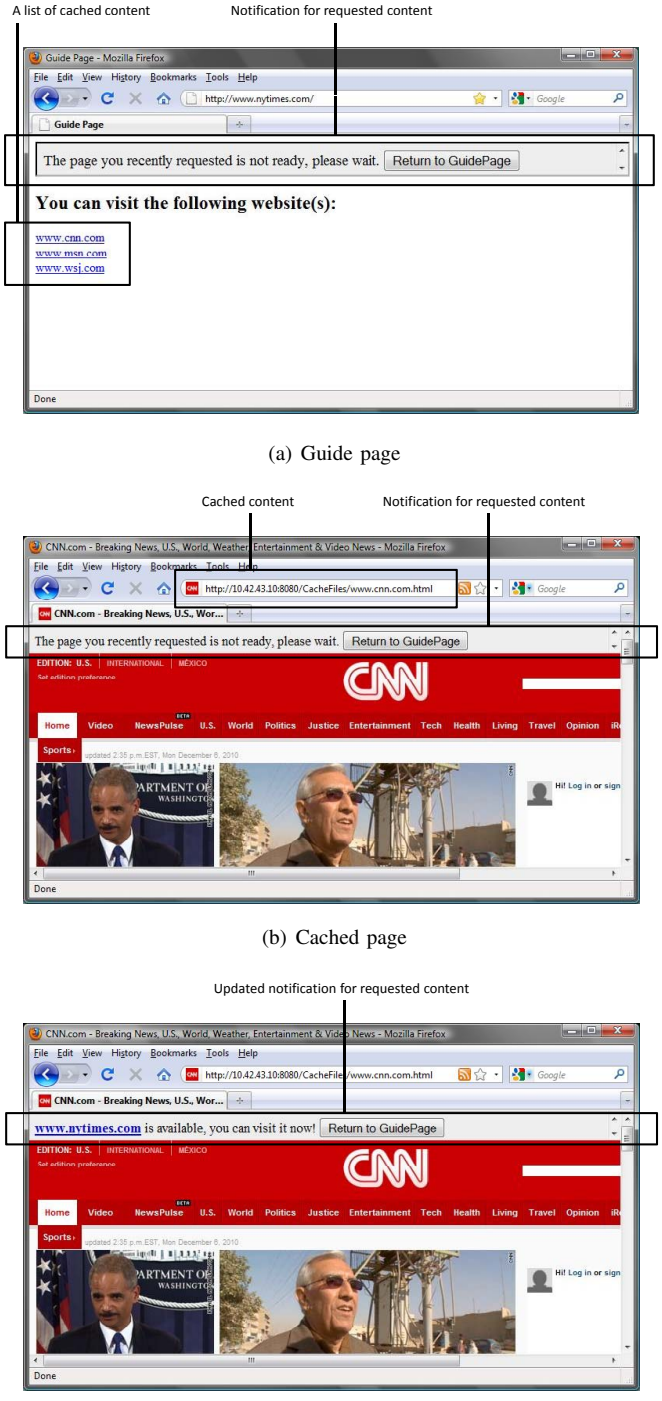
We use the OPNET modeler network simulator [3] to evaluate the performance of the proposed ICOW system compared with the direct access system (in which passegers access APs directly) in terms of throughput. Figs. 13(a) and (b) show the simulation topology of direct access system and ICOW, respectively.

Assumptions that we use in this simulation are:

- each passenger retrieves different web sites to avoid the case that a single web server handles all the requests,

[2]WPAD protocol is used to locate the corresponding URL of the PAC file (which is in the local web server in our system) using DHCP and/or DNS discovery method. Passengers can fetch the PAC file through this corresponding URL. Most web browsers, such as Internet Explore 8 and 9, FireFox 5.0, Chrome 12, and Safari 5, support WPAD.

[3]http://www.w3.org/TR/html5/. The HTML `iframe` represents a nested browsing context (inline frame) that allows another HTML document to be embedded within the current HTML document.



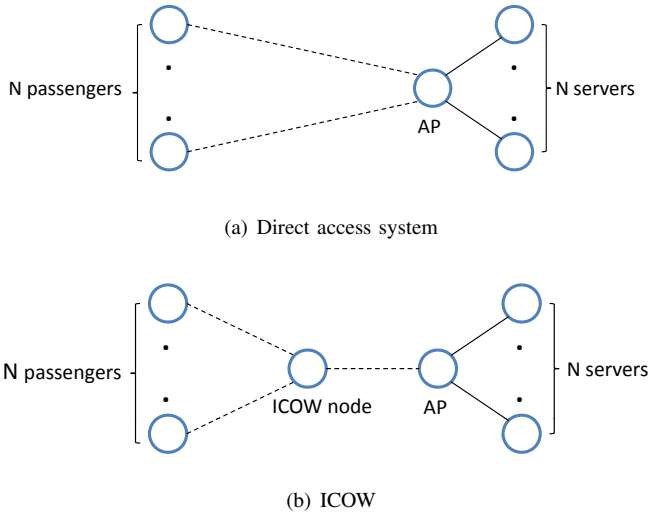
(a) Guide page

(b) Cached page

(c) Notification on the `iframe`

Fig. 12. Screen shots of a guide page and cached pages with `iframe` for notification

- we increase passenger's arrival rate (26/120, 30/120, 36/120 arrival/sec) according to Poisson distribution,
- each passenger generates a single HTTP request to a corresponding web server at randomly chosen [0-120] seconds of an entire simulation time where the first half is a network disconnection period [0-60] secconds while the last half is a network connection period [60-120] second,
- the generated HTTP requests during network disconnection period are queued and transmitted as soon as the network connection is available, at the beginning of network connection period (at 60 second of simulation time),
- using an image-based web browsing profile that generally has many images (i.e., a front page of many web sites follows the image-based web browsing profile), we select an HTTP request size of 100 KB and HTTP response size of 1 MB, including all inline requests and responses for the inline links of the page,
- and the channel rate for wireless links is 11 Mb/s.



(a) Direct access system

(b) ICOW

Fig. 13. Simulation topology. Dotted line is wireless network connection and solid line is wired connection.

Fig. 14 depicts the average throughput of ICOW, compared to the direct access system as a function of number of contending passenger nodes (N). The throughput measured by simulation only involves IEEE 802.11 DATA frames other than control and management frames. The simulation result shows that the average downlink (DL) throughput of the ICOW system is higher than that of the direct access system regardless of N. ICOW's throughput gain against the direct access system comes from that the ICOW nodes perform less backoff processes than the nodes in the direct access system. For $N = 35$, when the wireless link is about to be saturated, the ICOW system is able to obtain about 5 Mb/s throughput which is about 20 % better than the throughput (4 Mb/s) of the direct access system. On the other hand, the average uplink (UL) throughput for both systems is almost the same because

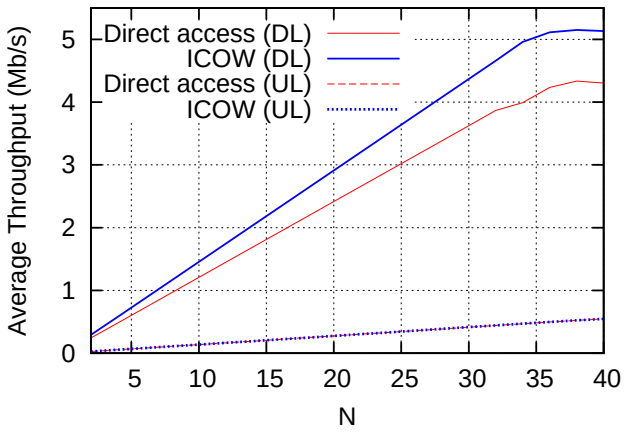


Fig. 14. Average throughput. N is the number of passengers. DL is downlink and UL is uplink. Channel rate between a bus and an AP is 11 Mb/s.

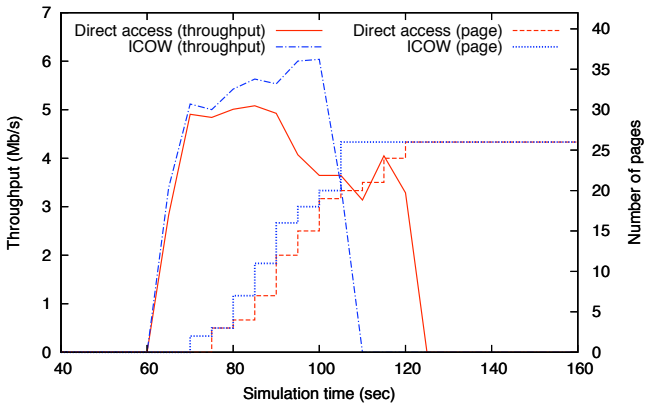
the relatively small size of HTTP requests (100 KB) affects the link throughput less.

Fig. 15 plots the downlink throughput of HTTP responses from an AP to passengers as well as the number of web pages that have been downloaded and cached for N=26, 30 and 36, respectively. The result shows that ICOW has a higher number of downloaded web pages than the direct access system within a given time. As shown in Fig. 15(a), when N=26, both ICOW and direct access system can download all HTTP responses before the network connection period ends. In Fig. 15(b), when N=30, all requested web pages are completely downloaded for 60 seconds in ICOW system while only 25 web pages are downloaded in a direct access system. When N=36, however, neither system can receive all web pages during the network connection period.

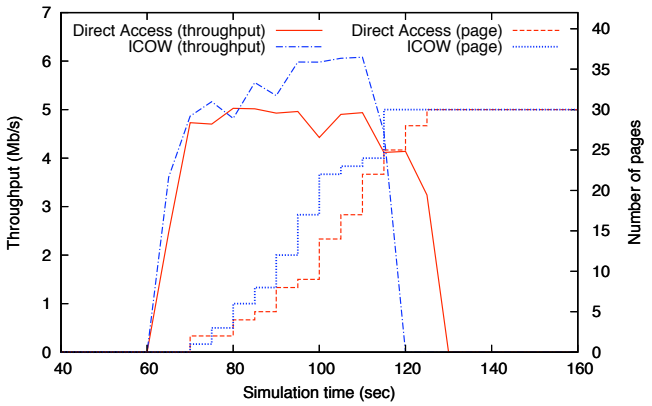
Another interesting result of this simulation is that at 90 seconds, as the number of passengers increases, the number of downloaded pages decreases. There are two reasons: the first reason is that the higher number of HTTP requests consumes more bandwidth so the HTTP responses are delayed. The other reason is that the limited bandwidth is shared by passengers to receive responses. ICOW can schedule the order of retrieval (e.g., serializing page download or limiting the number of simultaneous downloading pages). However, the direct access system has to share the bandwidth because there is no scheduling functionality in the system, so it suffers from longer delay to download a content as the number of requests are higher. Hence, there is larger number of incomplete web pages within a given period.

B. Cache hit ratio

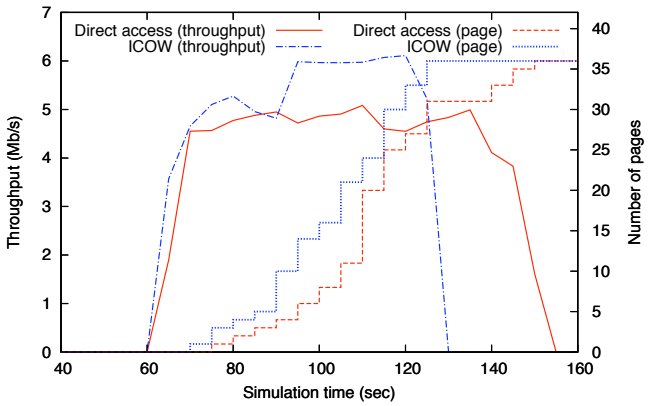
The cache hit ratio $Q(N_c, T)$ is the probability that a requested content after the T-th stop is in a cache when there



(a) N = 26



(b) N = 30



(c) N = 36

Fig. 15. Downlink throughput. Channel rate is 11 Mb/s. N is the number of passengers. The network disconnection period is from 0 second to 60 second. We keep network connection after 60 seconds in order to see when downloading all the requested web pages is finished.

are total N_c kinds of web pages.

$$Q(N_c, T) = \Pr(1_R(T) = 1) = \sum_{k=1}^{N_c} \Pr(R_K = k)\Pr(1_{R_k}(T) = 1),$$

where $1_R(T)$ is the indicator function that shows whether the requested content has been cached after the T-th stop or not. We assume that the random variable R_K is a Zipf random variable, and an independent and identically distributed random variable. Hence, $\Pr(R_K = k)$ is the probability mass function (pmf) of Zipf distribution: $(1/k^s)/\sum_{n=1}^{N_c}(1/n^s)$ where k is their popularity rank and s is the exponent characterizing Zipf distribution.

$\Pr(1_{R_k}(T) = 1)$ is the probability that content whose rank is k has been cached by the T-th stop. We N_r is the number of new requests at a stop, and we assume that it is a poisson random variable.

$$\begin{aligned}\Pr(1_{R_k}(T) = 1) &= 1 - \Pr(1_{\mathrm{R_k}}(\mathrm{T}) = 0) \\ &= 1 - \prod_{t=1}^{T} \Pr(1_{r_k}(t) = 0) \\ &= 1 - (\sum_{j=0}^{\infty} \Pr(N_r = j)(1 - \frac{1/k^s}{\sum_{n=1}^{N} 1/n^s})^j)^T,\end{aligned}$$

where $1_{R_k}(T)$ is the indicator function that shows whether the content whose rank is k has cached by the T-th stop and $1_{r_k}(t)$ is the indicator function that shows whether the web page whose rank is k is cached during the t-th stop.

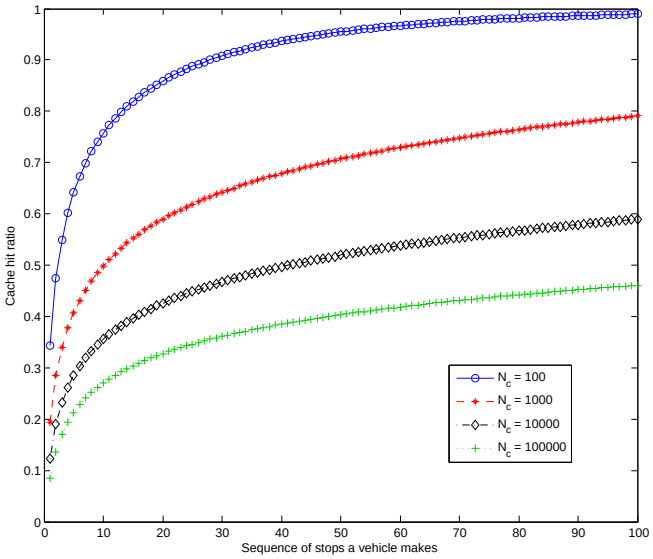


Fig. 16. Cache hit ratio. The passenger arrival rate λ is 10 per stop, and the exponent characterizing Zipf distribution s is one. N_c is the number of web pages.

Fig. 16 shows the cache hit ratio of web pages requested by passengers as a vehicle makes stops. The cache hit ratio increases as a vehicle makes more stops because the cached

web pages have been accumulated. This means that passengers are able to receive requested web pages from the local cache in an ICOW node, not from the Internet, with higher probability as a vehicle makes more stops. Hence, the bandwidth usage of the ICOW node during network connection periods decreases.

C. Request aggregation

In this Section, we evaluate a request aggregation mechanism. Expected number of requests in a queue $E[N_R]$ is the number of requests at the T-th stop with N_c web pages when we use request aggregation mechanism.

$$E[N_R] = \sum_{k=1}^{N_c} P_Q(R_k, T) = \sum_{k=1}^{N_c} \sum_{n=0}^{\infty} \Pr(N_r = n) P_Q(R_k, T, N_r)$$

where $P_Q(R_k, T)$ is the probability that a web page whose rank is k is in a queue at the T-th stop. We assume that the frequency of a web page whose rank is k is a Zipf random variable and independent and identically distributed random variable. The number of all requests from passengers N_r is a poisson random variable and $P_Q(R_k, T, N_r)$ is the probability that a web page whose rank is k is in a queue when there are N_r requests at the T-th stop.

$$P_Q(R_k, T, N_r) = \Pr(1_{Q_k}(N_r, T) = 1) \Pr(1_{R_k}(T) = 0)$$

where $1_{Q_k}(N_r, T)$ is the indicator function that shows whether the web page type k is requested at stop T when there are total N_r requests and $1_{R_k}(T)$ is the indicator function that shows whether the web page whose rank is k has cached by the T-th stop.

$$\begin{aligned} \Pr(1_{Q_k}(N_r, T) = 1) &= 1 - \Pr(1_{Q_k}(N_r, T) = 0) \\ &= 1 - (1 - \Pr(R_k = k))^{N_r} \\ &= 1 - (1 - \frac{1/k^s}{\sum_{n=1}^{N} 1/n^s})^{N_r}. \end{aligned}$$

$$\begin{aligned} \Pr(1_{R_k}(T) = 0) &= \prod_{t=1}^{T} \Pr(1_{r_k}(t) = 0) \\ &= (\sum_{j=0}^{\infty} \Pr(N_r = j)(1 - \frac{1/k^s}{\sum_{n=1}^{N} 1/n^s})^j)^T \end{aligned}$$

where $1_{r_k}(t)$ is the indicator function that shows whether the web page whose rank is k is cached during the t-th stop.

Fig. 17 shows the expected number of web pages that are downloaded from the Internet at each stop. We evaluate three mechanisms: direct access, ICOW with cache ($ICOW_{CO}$), and ICOW with request aggregation ($ICOW_{RA}$). In $ICOW_{CO}$, the downloaded web pages are cached, but request aggregation is not used. In $ICOW_{RA}$, the downloaded web pages are cached, and request aggregation mechanism is used. We get the expected number of pages of $ICOW_{CO}$ mechanism from the formula in Section VII-B. Both $\mathrm{ICOW_{CO}}$ and $\mathrm{ICOW_{RA}}$ reduce the number of pages, compared to the direct access mechanism. Because it eliminates duplicated requests and responses in the request aggregation mechanism, $\mathrm{ICOW_{RA}}$ has the smallest number of downloaded web pages. This means that the $ICOW_{RA}$ mechanism uses less bandwidth to handle the same requests from passengers.

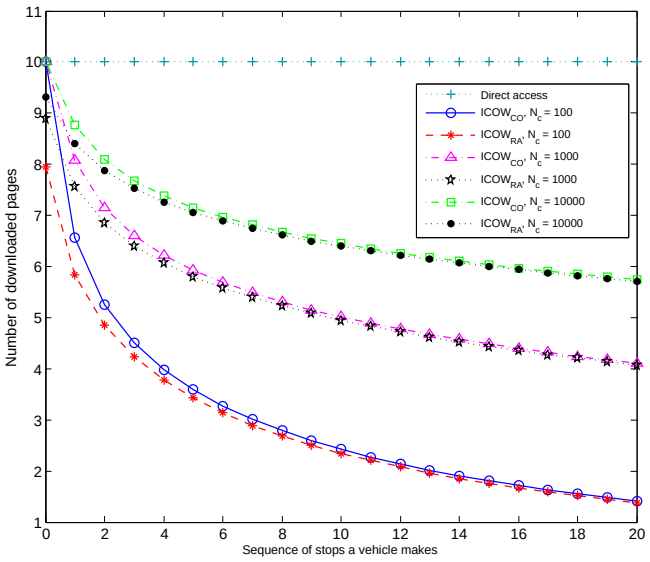


Fig. 17. Expected number of downloaded web pages at each stop. The passenger arrival rate λ is 10 per stop. N_c is the number web pages, and the exponent characterizing Zipf distribution s is one. $\mathrm{ICOW_{CO}}$ means that only caching is used, so request aggregation mechanism is not applied. $\mathrm{ICOW_{RA}}$ means that request aggregation is applied.

D. Scheduling algorithm

To evaluate scheduling algorithms, we applied two metrics: average content delivery delay and average aggregated utility. Utility function represents the value of users' satisfaction in completing jobs over time. C. B. Lee et al. [19] discussed that "this utility function is widely used in economics, and has recently been used in high-performance computing (HPC) and grid scheduling." In our system, the job is considered complete when passengers receive their requested content.

C. B. Lee et. al [19] conducted a survey to examine types of utility functions for real jobs on a real HPC system. They obtained three types of utility functions from the survey: step function, linear decay function, and exponential decay function. Fig. 18 shows the three utility functions. We applied these three types of utility functions to our evaluation.

The average aggregate utility, $V_i(t)$, of web page i is

$$V_i(t) = \frac{\sum_{j=1}^{N_i(t)} U_{ij}(t)}{N_i(t)},$$

TABLE I
PARAMETERS IN MONTE CARLO METHOD SIMULATION. THE NUMBER OF WEB PAGES IS 100,000. THE NUMBER OF STOPS IS 50. THE NUMBER OF SIMULATION IS 200.

	Passenger arrival (passengers/stop)	Passenger travel time (stops)	Rate of handling requests at each stop	Popularity of web pages	Content types (utility function)
Distribution	Poisson	Geometric	Uniform	Zipf	Discrete uniform
Range	mean: [10, 50]	mean: [4, 10]	[0.5, 1.0]	uniform distribution of s: [0, 1]	[step, linear decay, exponential decay]

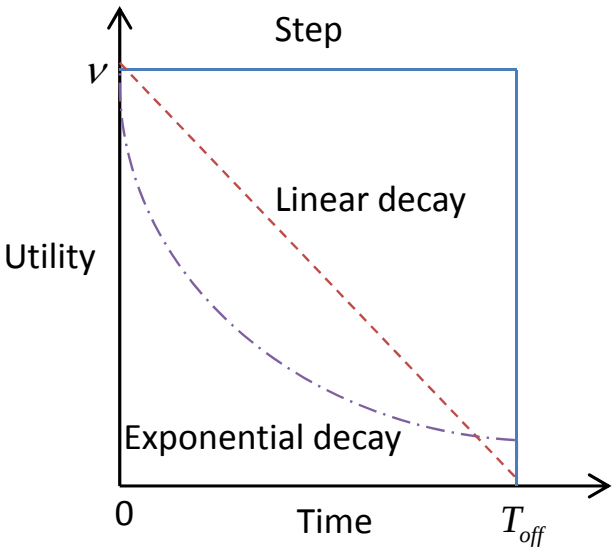


Fig. 18. Utility function $u(t)$. ν is a max utility value and T_{off} is the time that a passenger gets off.

where $U_{ij}(t)$ is the utility of content i of passenger j at time t and $N_i(t)$ is the number of requests for content i at time t.

We use the Monte Carlo method in order to evaluate performance due to the complexity of analysis. Shown in Table I, the parameters that we use in our analysis are passenger arrival rate, passenger travel time, the successful rate of handling requests, the number of web pages, popularity of web pages, and the utility function of web pages. We assume that each passenger requests one web page (including HTML fiels, images, CSS files, JavaScript files) while boarding the bus or train. We use a discrete time value, so the time value n means the n th stop from a vehicle departs from its depot. If passengers request web pages at the same stop, the generation times of those requests are the same.

We evaluate five scheduling algorithms: FCFS, FCFS with popularity for tie-breaker (FCFS+popularity), popularity with FCFS for tie-breaker (Populairty+FCFS), popularity with most-recent-first (MRF) for tie-breaker (Populairty+MRF), and maximizing utility with MRF for tie-breaker (Max utility+MRF).

According to Table II, Max utility+MRF has the best performance, as we expected. This scheduling algorithm is ideal because it reflects utility functions of all the requests to maximize satisfaction of passengers. However, it is based on the strong assumption that this algorithm knows the utility value of each request. In real implementation, however, it is almost impossible to satisfy this assumption.

Our goal is to propose an algorithm that has as close performance as Max utility+MRF while the proposed algorithm supports easy implementation. Table II shows that our proposed scheduling algorithm, Popularity+MRF, has close performance results of Max utility+MRF in terms of average aggregate utility and delay. Because of MRF function of Popularity+MRF, it has similar average aggregate utility value as Max utility+MRF. The utility has the highest value at the beginning of the requesting time and decays as the time flows. Because of MRF function, those two algorithms have a shorter delivery delay than others. However, popularity-based algorithm can be easily implemented by checking the number of requests at an ICOW node. Thus, considering the complexity of the implementation, Popularity+MRF might be suitable in terms of easiness of implementation and performance.

VIII. ACKNOWLEDGMENT

This work is supported by a grant from InterDigital Communications, LLC.

IX. CONCLUSION

To maximize content availability and system usability in intermittently connected networks for public transit systems, we propose ICOW. This system is a cache- and proxy-based system to enhance link throughput and resolve network resource limitations. Not only does this allow for maximizing network resource utility and minimizing bandwidth usage, but it also provides ease of system use for passengers.

Our evaluation shows that ICOW increases link throughput compared to a direct access system. Our caching-based mechanism allows more passengers to get more content with limited network resources, and decreases the delay of getting content. By using popularity-based scheduling mechanism, ICOW provides higher satisfaction of obtaining content for passengers and decreases content delivery delay.

REFERENCES

[1] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on selected areas in communications*, vol. 18, no. 3, pp. 535–547, 2000.

[2] R. P. Liu, G. J. Sutton, and I. B. Collings, "A new queueing model for QoS analysis of IEEE 802.11 DCF with finite buffer and load," *IEEE Transactions on Wireless Communications*, vol. 9, no. 8, pp. 2664–2675, 2010.

[3] OPNET modeler, http://www.opnet.com.

TABLE II
MONTE CARLO METHOD SIMULATION RESULTS OF MESSAGE SCHEDULING. MAIN ALGORITHM + TIE-BREAKER. MRF STANDS FOR MOST-RECENT-FIRST.

Algorithm	FCFS	FCFS + Popularity	Popularity + FCFS	Popularity + MRF	Max utility + MRF
Average aggregate utility	0.67	0.67	0.68	0.77	0.82
Average delivery delay (stops)	2.10	2.10	2.10	1.19	1.15
Number of passengers who receive content	21.89	21.89	21.90	21.98	22.09

[4] S. Srinivasan, A. Moghadam, S. G. Hong, and H. Schulzrinne, "7DS - Node Cooperation and Information Exchange in Mostly Disconnected Networks," in *Proceedings of IEEE International Conference on Communications (ICC)*, Glasgow, Scotland, June 2007.

[5] S. Pack, H. Rutagemwa, X. Shen, J. Mark, and K. Park, "Efficient Data Access Algorithm for ITS-based Networks with Multi-Hop Wireless Links," in *Proceedings of IEEE International Conference on Communications (ICC)*, Glasgow, Scotland, June 2007.

[6] A. Balasubramanian, B. N. Levine, and A. Venkataramani, "Enhancing interactive web applications in hybrid networks," in *Proceedings of the ACM international conference on Mobile computing and networking (MobiCom)*, San Francisco, California, USA, September 2008.

[7] J. Ott and D. Kutscher, "Bundling the web: HTTP over DTN," in *Proceedings of Workshop on Networking in Public Transport (WNEPT)*, Waterloo, Ontario, Canada, August 2006.

[8] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic Routing in Intermittently Connected Networks," *SIGMOBILE Mobile Computing and Communications Review*, vol. 7, pp. 19–20, July 2003. [Online]. Available: http://doi.acm.org/10.1145/961268.961272

[9] J. Leguay, T. Friedman, and V. Conan, "DTN routing in a mobility pattern space," in *Proceedings of the ACM SIGCOMM workshop on Delay-tolerant netwoking (WDTN)*, Philadelphia, PA, August 2005.

[10] P. Hui, J. Crowcroft, and E. Yoneki, "BUBBLE RAP: Socail-based Forwarding in Delay Tolerant Networks," in *Proceedings of ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Hong Kong SAR, China, May 2008.

[11] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant MANETs," in *Proceedings of the ACM international symposium on Mobile ad hoc networking and computing (Mobihoc)*, Montreal, QC, Canada, September 2007.

[12] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "MaxProp: Routing for Vehicl-Based Disruption-Tolerant Networks," in *Proceedings of IEEE INFOCOM*, Barcelona, Spain, April 2006.

[13] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, "Hypertext Transfer Protocol – HTTP/1.1," Internet Engineering Task Force: RFC 2616, June 1999.

[14] Muffin: World Wide Web Filtering System, http://muffin.doit.org/.

[15] A. Luotonen, "Navigator Proxy Auto-Config File Format," Netscape Corporation, March 1996.

[16] P. Gauthier, J. Cohen, M. Dunsmuir, and C. Perkins, "Web proxy auto-discovery protocol," Internet Engineering Task Force, Internet Draft, December 1999, work in progress.

[17] Apache Tomcat, http://tomcat.apache.org/.

[18] Ajax push, http://www.icefaces.org/main/ajax-java/ajaxpush.iface.

[19] C. B. Lee and A. E. Snavely, "Precise and Realistic Utility Functions for User-Centric Performance Analysis of Schedulers," in *Proceedings of IEEE International Symposium on High Performance Distributed Computing (HPDC)*, Monterey, California, USA, June 2007.