On the Analysis of Cache Invalidation With LRU Replacement

Quan Zheng, Tao Yang, Yuanzhi Kan, Xiaobin Tan, Jian Yang, *Senior Member, IEEE*, and Xiaofeng Jiang, *Member, IEEE*

Abstract—Caching contents close to end-users can improve the network performance, while causing the problem of guaranteeing consistency. Specifically, solutions are classified into validation and invalidation, the latter of which can provide strong cache consistency strictly required in some scenarios. To date, little work on the analysis of cache invalidation has been covered. In this work, by using conditional probability to characterize the interactive relationship between existence and validity, we develop an analytical model that evaluates the performance (hit probability and server load) of four different invalidation schemes with LRU replacement under arbitrary invalidation frequency distribution. The model allows us to theoretically identify some key parameters that affect our metrics of interest and gain some common insights on parameter settings to balance the performance of cache invalidation. Compared with other cache invalidation models, our model can achieve higher accuracy in predicting the cache hit probability. We also conduct extensive simulations that demonstrate the achievable performance of our model.

Index Terms—LRU, cache, consistency, invalidation, hit probability, server load

1 Introduction

Cache has been extensively employed to enhance the network performance. Benefiting from cache hits, bandwidth usage over links, user-perceived delays and loads on the origin server are reduced appreciably [1]. Information-Centric Networking (ICN) paradigm has emerged for efficient content distribution, and various candidate architectures including Named Data Networking (NDN) or Content Centric Networking (CCN), Publish/Subscribe Internet Routing Paradigm (PSIRP), and Network of Information (NetInf) have been introduced. In NDN, every relayed content is stored in the cache of routers, and it is used to serve future requests [2]. Ubiquitous caching is one of the most distinctive features in this novel architecture for future network.

Owing to the introduction of caching, cache consistency also has constantly attracted the attention of many researchers [3], [4], [5], [6], [7], [8], [9], [10], [11]. Caching generates numerous copies of contents distributed throughout the network. If, for example, a master content is updated at the origin server, the copies of that content stored at the caches may become outdated. Hence, maintaining cache consistency is necessary to ensure the copies obtained by users are valid.

There are two underlying approaches for the cache consistency: validation and invalidation [12]. With validation, the caches verify the validity of their stored contents with the origin server periodically and therefore this approach guarantees weak consistency only. However, some applications, such as financial transactions, require strong consistency which can only be provided by invalidation. Moreover, cache invalidation can be classified into four basic schemes:

Reactive Invalidation. When a request for a content arrives and there is a corresponding copy existing in the cache, the cache then sends an *If-Modified-Since* request to the server, which in turn will reply either with a *304 Not-modified* message if the copy is deemed up to date, or with the latest full data if the copy is stale [13].

Proactive Invalidation With Removing. When a master content is updated at the server, the server notifies caches of the change. After receiving notifications, the caches remove the stale copy [12].

Proactive Invalidation With Renewing. When a master content is updated at the server, the server pushes the latest copy to the caches which have stored the stale one [14].

Proactive Invalidation With Optional Renewing. Unlike the second and third schemes, the notification sent by the server consists of either an invalidation message or a updated copy in this scheme. For avoiding the waste that the updated copy is never subsequently requested at the caches, usually only the copies of popular contents are renewed, while the copies of unpopular ones are removed [15].

Formally speaking, most Web applications apply validation rather than invalidation to maintain cache consistency due to the extra overhead on the network caused by the latter [12]. Concretely, for reactive invalidation, each cache hit

• *Quan Zheng is with the Department of Automation, University of Science and Technology of China, Hefei 230000, China, and also with the Institute of Advanced Technology, University of Science and Technology of China, Hefei 230088, China. E-mail: qzheng@ustc.edu.cn.*
• *Tao Yang, Yuanzhi Kan, Xiaobin Tan, Jian Yang, and Xiaofeng Jiang are with the Department of Automation, University of Science and Technology of China, Hefei 230000, China. E-mail: {yt2015, kan}@mail.ustc.edu.cn, {xbtan, jianyang, jxf}@ustc.edu.cn.*

Manuscript received 12 July 2020; revised 8 July 2021; accepted 12 July 2021. Date of publication 21 July 2021; date of current version 11 Aug. 2021. (Corresponding author: Xiaobin Tan.) Recommended for acceptance by S. Pakin. Digital Object Identifier no. 10.1109/TPDS.2021.3098459

will trigger a verification process, the number of which increases sharply with network scale expanding. For proactive invalidation, the server has to maintain per-content status consisting of a list of all caches that have the copies of this content. As the number of contents enlarges and the copies are more widely distributed throughout the network, such status could become too enormous to maintain. Therefore, how to reduce above overhead is the most imperative issue to be addressed when deploying the cache invalidation. In particular, the forth scheme, proactive invalidation with optional renewing, can reduce the content status overhead and the number of invalidation messages by only maintaining the status of specific contents at the server [15]. Nevertheless, how many specific contents should be maintained to achieve a decrease of the overhead without appreciably affecting the network performance is another issue that needs to be addressed.

Unfortunately, most of the current work of cache invalidation is focused on the specific strategies [7], [8], [9], [10], [16]. Little work has been proposed on establishing theoretical models to analyze the cache invalidation quantitatively. In this paper, we propose an analytical model to evaluate the performance of above four invalidation schemes with LRU replacement. The frequency of content update can follow any distribution in our model. Additionally, in order to estimate the overhead that is extra but necessary to guarantee the strong cache consistency, we take the actively renewing load into the consideration of total server load. Furthermore, our model takes great advantages on extensibility and accuracy compared with other cache invalidation models.

The major contributions of our work are:

- We model four invalidation schemes with LRU replacement by using conditional probability to characterize the interactions between existence and validity under arbitrary invalidation frequency distribution. In our analysis, we compute the hit probability and server load.
- We use our model to estimate the performance of four invalidation schemes, and identify some key parameters that affect their performance. Particularly, we theoretically give the point of the highest hit probability for the proactive invalidation with optional renewing. In addition, we generalize some principles of parameter settings to balance the hit probability and server load.
- We perform numerical simulations to demonstrate that our model can achieve high accuracy in predicting the hit probability and server load for individual content requests. We also apply our model to realistic traffic data to estimate the invalidation effect on the cache hit probability.

The remainder of this paper is organized as follows. In the next section, we review the related work. In Section 3, we describe the invalidation model and discuss some basic assumptions. Section 4 models the four invalidation schemes and gives analytical expressions respectively. Section 5 evaluates the accuracy of our model and presents the simulation results. Finally, conclusions and future work are given in Section 6.

2 Related Work

Thanks to the significance of caching to the web applications, improving the caching performance has always been a hot topic in the networking research community for decades. Thereinto, a considerable amount of works on replacement policies for a single cache were produced, from classical LRU, LFU, FIFO and RANDOM algorithms to modified LRU-K [17], LFRU [18] and many other ones. However, the requirement for line-rate operation limits the complexity of the cache replacement policy. Thus most of the cache systems commonly apply LRU algorithm that is easy to be implemented as well as has a complexity of $O(1)$ [1].

2.1 Evaluating the Performance of LRU Cache

Many analytical models for evaluating the performance of LRU cache also have been established in these years. The first exact model was provided by King [19] to estimate the hit probability of the cache. Flajolet *et al.* [20] then gave a simpler expression. Regrettably, the computational overhead of [19] and [20] grows exponentially with the cache capacity and the number of contents increasing. Almost at the same period as [20], Dan *et al.* [21] developed a simple approximate analytical model with the computational complexity of quadratic time. About 20 years later, Rosensweig *et al.* [22] developed a novel model for general-topology cache networks by utilizing the work in [21].

With the development of networking, cache network is of a more massive scale and an efficient model to evaluate the caching performance is urgently needed. A more widely used model was proposed by Che *et al.* [23] in 2002, originally known as the characteristic time approximation. The analytical model in [23] provides extremely accurate results at low computational overhead under the traffic of Independent Reference Model (IRM for short), and a theoretical explanation for the success of this model is given in [24].

Based on the characteristic time approximation, Martina *et al.* [25] proposed a unified methodology to analyze the performance of cache systems. The study in [25] extends the traffic condition relied on in [23] from IRM to renewal traffic, leading to much broader applicability.

The above research works provide fundamental models that can be expanded to some specific cache scenarios, such as cache consistency.

2.2 Cache Consistency and Analytical Models

The problem of cache consistency is another extensively studied topic in networking, especially with the emergence of CCN. As we mentioned earlier, the approaches for cache consistency can be divided into validation and invalidation.

Cache validation provides only weak consistency. The basic validation scenario is the Time-to-live (TTL) cache where each content is associated with an expiration time of a constant [3] or a variable parameter [4]. The first request presented after the expiration time is forwarded to the server, which successively sends a valid copy to the cache. Analytical models for TTL-based cache were given in [5], [6]. The problem of placing replicas under the widely used TTL-based consistency scheme has been investigated [26]. Freshness and Timestamp are also common technologies for validation in cache networks. Amadeo *et al.* [27] added a timestamp to the data packets to facilitate the evaluation of their freshness, so as to decide whether to remove them from the cache. Vural *et al.* [28] proposed an analytical model that captures the trade-off between multi-hop communication costs and data item freshness.

In contrast, cache invalidation, commonly classified into the reactive and the proactive, can provide strong consistency. Several invalidation strategies have been proposed such as Leases [7] for distributed file system, Piggyback server invalidation [8], IR-based cache invalidation [9], Bit-Sequences [10] for mobile environments, cache invalidation strategies in wireless environments [16], etc.

There is little work on modeling the cache invalidation until 2018 that Detti *et al.* [11] proposed an analytical model to evaluate the performance of LRU cache that takes into account of reactive invalidation and proactive invalidation with removing. In the Detti model, the invalidation event is considered as an independent random process. While in our model, the invalidation event is associated with the existence event, and these two events are regarded as a pair of interactions that are characterized by conditional probability. For computing the hit probability of proactive invalidation with removing, both Detti model and our model can achieve a very high accuracy. However, Equations (5) and (6) presented in [11] for computing the hit probability of reactive invalidation, indeed, ignore the fact that the requests for outdated copies stored in cache can not be regarded as a hit event. The hit probability of reactive invalidation obtained from these two equations is identical to the hit probability without invalidation, which has been validated in our comparison results in Section 5. Besides, the Detti model can not be developed to compute the hit probability of proactive invalidation with optional renewing that is able to enable a better performance than other invalidation schemes [15].

The model proposed in our work is more extensible than the Detti model, and can achieve a higher accuracy on predicting the hit probability of reactive invalidation. Moreover, to the best of our knowledge, we are the first to propose an accurate analytical model to estimate the performance of proactive invalidation with optional renewing under a few assumptions which are commonly applied in cache modeling and presented in the next section.

3 Model Description

3.1 Assumptions and Notations

Using the notations described in Table 1, we make the following assumptions:

- We consider a collection of M same size contents. Without loss of generality, the size of each content is equal to 1 and therefore the cache capacity C and occupancy size O can be measured in the unit of content quantity [23]. For some contents of different sizes, we can divide them into constant sized chunks [24], [29]. The modeling process is only related to the characteristic time, and has nothing to do with the content size [23]. We assume that the contents of the size of 1 in order to simplify the calculation.
- Let $\{R_i^m\}_{i=1}^{\infty}$ be a sequence of request arrivals for content m. The request probability of content m is α^m which is constant and independent of all past requests. It corresponds to the IRM widely used in many studies of network cache [19], [20], [21], [23]. In our work, we adopt a Zipf law to model α^m

$$\alpha^m = \frac{K}{R(m)^z}, \tag{1}$$

where $K = 1/\sum_{m=1}^{M} \frac{1}{R(m)^z}$ is the Zipf normalization factor, $R(m) = 1, 2, \ldots, M$ is the popularity rank of content m, and the exponent z is a parameter ranging between 0.65 and 1 generally.
- User requests for all contents arrive according to a Poisson process with mean arrival rate λ. Thus, the requests for content m also arrive according to a Poisson process with mean arrival rate $\lambda^m = \alpha^m \lambda$ based on the above assumptions.
- In order to simplify the analysis, we do not consider the delay in data transmission. In other words, if a cache miss happens, a new content copy will be instantly downloaded from the server. It should be noted that the data transmission delay will have an impact on the probability of the request hitting the valid content copy. We discuss this impact in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2021.3098459.

TABLE 1
Notations

Notation	Meaning
C, O	Cache capacity and cache occupancy size
L	Server load
M	The number of different contents
$N_u^m(t)$	The number of updates for content m by time t
$N_r^m(t)$	The number of requests for content m by time t
α^m	Request probability of content m
λ, λ^m	Request rate of total contents and content m
P_e, P_e^m	Mean existence probability of total contents and content m
P_h, P_h^m	Mean hit probability of total contents and content m
P_v, P_v^m	Mean validity probability of total contents and content m
$P_{e,v}^m$	The joint probability of existence and validity for content m
$P_{e\|v}^m$	The conditional probability of existence given validity for content m
$P_{v\|e}^m$	The conditional probability of validity given existence for content m
T_c	Cache characteristic time
T_s^m, T_u^m	Expiration time and update time of content m
μ_s^m	Expected value of the expiration time for content m
$F(t_s^m)$	CDF of the expiration time for content m
$F_r(i, t)$	CDF of the content arriving time in the ith interval
$Y(m)$	The number of content m's copy stored in the cache
Y	The number of all contents' copies stored in the cache
M_p	The top M_p popular contents

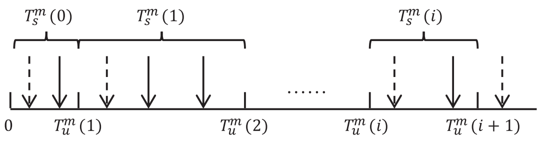


Fig. 1. Requests for content m, where the dotted arrows yield *invalidity* and the solid arrows yield *validity*.

- Since the size of the invalidation message in the interactions between the server and cache is tinier than that of the data packet, we ignore the former impact on the server load.

3.2 The Characteristic Time Approximation

LRU cache can be regarded as a stack. When a content request arrives and gets a cache hit, the content copy in cache will be reinserted into the top of the stack. Otherwise, a cache miss occurs and a new content copy will be downloaded from the server and inserted into the top of the stack. If the stack size reaches the limit, a content copy at the bottom of the stack will be evicted (We call this event *eviction*). The time that a content copy moves from the top to the bottom without any cache hits taking place and finally is evicted from the stack is described as the cache characteristic time T_c in the characteristic time approximation [23].

With the consideration of T_c, a content m's copy exists in the cache at time t, if and only if the last request for content m arrives at the cache in the interval $(t - T_c, t]$. By assuming that the requests for content m arrive according to a Poisson process with rate λ^m, the probability P_e^m that content m's copy is in the cache can be computed as

$$P_e^m = 1 - e^{-\lambda^m T_c}. \tag{2}$$

Let $Y(m)$ denote the number of content m's copy stored in the cache. Obviously, it takes value 1 if a content m's copy exists in the cache and 0 otherwise. Let Y denote the number of all contents' copies stored in the cache. Under the assumption that the size of each content is equal to 1, the expected number of all contents' copies stored in the cache is written as $E[Y] = E\left[\sum_{m=1}^M Y(m)\right] = \sum_{m=1}^M E[Y(m)] = \sum_{m=1}^M 1 \cdot P_e^m$. With regard to a stable cache system, the space of the cache is always full, and thus $E[Y] = C$. Then we can get a fixed point equation as follows:

$$\sum_{m=1}^{M} (1 - e^{-\lambda^m T_c}) = C. \tag{3}$$

The value of T_c can be obtained by calculating the unique root of this equation [23], [24].

3.3 Probability of Validity

In this section, we give a basic model to describe whether a content copy in cache is valid when a request for it arrives. In order to eliminate the influence of the cache capacity limit, we assume that the cache capacity is infinite and all contents' copies have been stored already.

As shown in Fig. 1, the server updates the content m according to a point process $\{T_u^m(i), i = 0, 1, 2, \ldots\}$ where $T_u^m(i)$ represents the time that the update takes place. Thus,

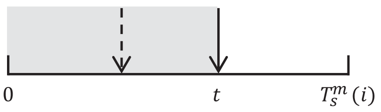


Fig. 2. Consider *validity* in a single interval.

we have $T_u^m(i) \le T_u^m(i+1)$ for each $i = 0, 1, 2, \ldots$ with $T_u^m(0) = 0$. Let $\{T_s^m(i), i = 0, 1, 2, \ldots\}$ represent an i.i.d. sequence with a common distribution $F(t_s^m)$. We shall interpret $T_s^m(i)$ as the content expiration time between $T_u^m(i)$ and $T_u^m(i+1)$, i.e., $T_s^m(i) = T_u^m(i+1) - T_u^m(i)$. In order to avoid trivialities, we also suppose that $T_s^m(i) \neq 0$, which means multiple updates for content m cannot occur simultaneously. Let

$$\mu_s^m = E[T_s^m(i)] = \int_0^\infty t_s^m dF(t_s^m), \tag{4}$$

denote the mean time between successive updates and $N_u^m(T)$ denote the number of updates for content m by time T. Then we have [30]

$$\lim_{T \to \infty} \frac{N_u^m(T)}{T} = \frac{1}{\mu_s^m}. \tag{5}$$

In each interval $T_s^m(i)$, the content m's copy hit by the first request (the dotted arrows in Fig. 1) is stale and we call this event *invalidity*. Then it is removed from the cache and a new copy is downloaded from the server. Therefore, the subsequent requests (the solid arrows) can obtain valid copies in this interval and we call this event *validity*.

Let us consider *validity* in a single interval first. Without loss of generality, the start time of the interval is supposed to be 0 as shown in Fig. 2. According to the above description, if a request arriving at t yields *validity*, there must be at least one request arriving in the grey area.

Assuming that the request arrival for content m is a Poisson process, the number of requests in any interval of length t is Poisson distributed with mean $\lambda^m t$. That is, for all $s, t \ge 0$

$$P\{N_r^m(t+s) - N_r^m(s) = n\} = e^{-\lambda^m t} \frac{(\lambda^m t)^n}{n!}, \tag{6}$$

where $N_r^m(t)$ represents the number of requests that have occurred up to time t and $N_r(0) = 0$. So the probability that the request for content m arriving at t yields *validity*in the ith interval can be expressed as

$$P_v^m(i,t) = 1 - P\{N_r^m(t) = 0\} = 1 - e^{-\lambda^m t}. \tag{7}$$

For Poisson process, the time t is uniformly distributed over $[0, T_s^m(i)]$

$$F_r(i,t) = \frac{t}{T_s^m(i)}. \tag{8}$$

Hence the expected value of $P_v^m(i,t)$ in the ith interval is

$$\begin{aligned} P_v^m(i) &= \int_0^{T_s^m(i)} P_v^m(i,t) dF_r(i,t) \\ &= \int_0^{T_s^m(i)} \frac{1}{T_s^m(i)} (1 - e^{-\lambda^m t}) dt. \end{aligned} \tag{9}$$

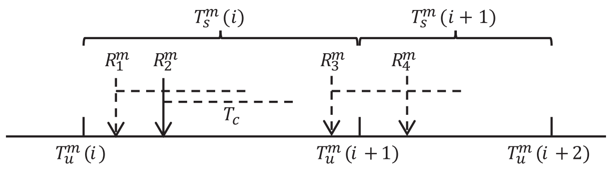


Fig. 3. Classification of the requests for content m, $i = 0, 1, 2, \ldots$

Now we reconsider *validity* with all intervals. The validity probability P_v^m can be expressed as the value of the weighted accumulation of $P_v^m(i)$. On the basis of (9), we have

$$\begin{aligned} P_v^m &= \lim_{T\to\infty} \sum_{i=0}^{N_u^m(T)} \frac{T_s^m(i)}{T} P_v^m(i) \\ &= \lim_{T\to\infty} \sum_{i=0}^{N_u^m(T)} \frac{1}{T} \int_0^{T_s^m(i)} (1 - e^{-\lambda^m t}) dt. \end{aligned} \tag{10}$$

Hence, on the basis of (5) and (10), the expected value of P_v^m can be expressed as

$$\begin{aligned} E[P_v^m] &= E\left[\lim_{T\to\infty} \sum_{i=0}^{N_u^m(T)} \frac{1}{T} \int_0^{T_s^m(i)} (1 - e^{-\lambda^m t}) dt\right] \\ &= \lim_{T\to\infty} \sum_{i=0}^{N_u^m(T)} \frac{1}{T} E\left[\int_0^{T_s^m(i)} (1 - e^{-\lambda^m t}) dt\right] \\ &= \lim_{T\to\infty} \frac{N_u^m(T)}{T} \int_0^{\infty} \int_0^{t_s^m(i)} (1 - e^{-\lambda^m t}) dt dF(t_s^m) \\ &= \frac{1}{\mu_s^m} \int_0^{\infty} \int_0^{t_s^m} (1 - e^{-\lambda^m t}) dt dF(t_s^m). \end{aligned} \tag{11}$$

The validity probability and hit probability mentioned below both refer to their respective expected values.

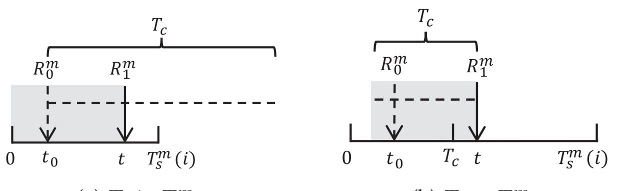
In the model assumptions, it is assumed that the data transmission delay is out of consideration. However, based on (7), (8), (9) and (11), the influence of transmission delay on the validity probability can be given as presented in Appendix A, available in the online supplemental material.

4 Cache Invalidation With LRU Replacement

In this section, we give a complete model with the consideration of both existence and validity to derive the cache hit probability and server load.

As illustrated in Fig. 3, the requests for content m are classified into four types:

- *non-existence and invalidity*: when R_1^m arrives, there is no copy of content m existing in the cache. Meanwhile, it is the first request in the interval $[T_u^m(i), T_u^m(i+1))$.
- *existence and validity*: the next request R_2^m arrives before the characteristic time T_c and is not the first request in $[T_u^m(i), T_u^m(i+1))$.
- *non-existence but validity*: the third request R_3^m is also not the first request in $[T_u^m(i), T_u^m(i+1))$ but arrives after T_c.
- *existence but invalidity*: the last request R_4^m arrives before T_c but is a first request in $[T_u^m(i+1), T_u^m(i+2))$.



(a) $T_c \geq T_s^m$ (b) $T_c < T_s^m$

Fig. 4. If there is a request arriving in grey area, R_1^m must yield an *existence and validity* event.

Thus, only the requests of type *existence and validity* can yield cache hits, otherwise cache misses take place. For content m, let us denote the probability of *existence and validity* in the ith interval as $P_{e,v}^m(i)$ where e represents the event *existence* and v represents the event *validity*. Then the hit probability of content m can be expressed as

$$P_h^m(i) = P_{e,v}^m(i) = P_{e|v}^m(i) \cdot P_v^m(i), \tag{12}$$

or

$$P_h^m(i) = P_{e,v}^m(i) = P_{v|e}^m(i) \cdot P_e^m, \tag{13}$$

where $P_{e|v}^m(i)$ and $P_{v|e}^m(i)$ are conditional probabilities. Then the average hit probability of content m is

$$P_h^m = E\left[\lim_{T\to\infty} \sum_{i=0}^{N_u^m(T)} \frac{T_s^m(i)}{T} P_h^m(i)\right], \tag{14}$$

and average hit probability of total contents is

$$P_h = \sum_{m=1}^{M} \alpha^m P_h^m. \tag{15}$$

Now we can model the four different schemes of cache invalidation separately and give the hit probability as well as server load based on (12), (13), (14) and (15).

4.1 Reactive Invalidation

Considering that different characteristic time T_c and content expiration time T_s^m have different effects on the analysis of $P_{e|v}^m$ and $P_{v|e}^m$, we model the cache behavior in two cases as follows:

4.1.1 $T_c > T_s^m(i)$

Assuming that a request R_1^m arrives at time t as shown in Fig. 4a. If it yields a *validity* event, there must be at least one request arriving in $[0, t)$ (the grey area). Let t_0 denote the time that the first request R_0^m arrives in $[0, t)$. Since $T_c > T_s^m(i)$ and $0 \leq t_0 < t < T_s^m(i)$, we have $t_0 < t < T_c$. That is to say, there must be a copy of content m existing in cache at time t and thus $P_{e|v}^m(i) = 1$. On the basis of (9) and (12), we have

$$P_h^m(i) = P_v^m(i) = \int_0^{T_s^m(i)} \frac{1}{T_s^m(i)} (1 - e^{-\lambda^m t}) dt. \tag{16}$$

4.1.2 $T_c \le T_s^m(i)$

As shown in Fig. 4b, we can separate the interval $[0, T_s^m(i))$ into two periods: $[0, T_c)$ and $[T_c, T_s^m(i))$. If the arrival time t of R_1^m is in $[0, T_c)$, the analysis is the same as that with $T_c > T_s^m(i)$. Otherwise, t is in $[T_c, T_s^m(i))$. If there is a copy of content m existing in cache at time t, there must be a request R_0^m arriving in $[t - T_c, t)$. That is to say, R_1^m must yield a *validity* event and $P_{v|e}^m = 1$. With the assumption that the requests arrive according to a Poisson process, the arrival time t of R_1^m is uniformly distributed. Therefore, the probability of $0 \le t < T_c$ is $T_c/T_s^m(i)$ and the probability of $T_c \le t < T_s^m(i)$ is $1 - T_c/T_s^m(i)$. Then the hit probability of content m can be written as (17) based on (2), (12), (13) and (16).

$$\begin{aligned}P_h^m(i) &= P_{0\le t<T_c} \cdot P_{(e,v)|0\le t<T_c}^m(i) + P_{T_c\le t<T_s^m(i)} \cdot P_{(e,v)|T_c\le t<T_s^m(i)}^m(i) \\ &= \frac{T_c}{T_s^m(i)} P_{(e|v)|0\le t<T_c}^m(i) \cdot P_{v|0\le t<T_c}^m(i) \\ &\quad + \frac{T_s^m(i) - T_c}{T_s^m(i)} P_{(v|e)|T_c\le t<T_s^m(i)}^m(i) \cdot P_{e|T_c\le t<T_s^m(i)}^m(i) \\ &= \frac{1}{T_s^m(i)} \int_0^{T_c} (1 - e^{-\lambda^m t}) dt + \left(1 - \frac{T_c}{T_s^m(i)}\right)(1 - e^{-\lambda^m T_c}).\end{aligned} \tag{17}$$

Consequently, according to (5), (14), (16) and (17), we have

$$\begin{aligned}P_h^m = &\frac{1}{\mu_s^m} \int_0^{T_c} \int_0^{t_s^m} (1 - e^{-\lambda^m t}) dt dF(t_s^m) \\ &+ \frac{1}{\mu_s^m} \int_{T_c}^{\infty} \left\{ \int_0^{T_c} (1 - e^{-\lambda^m t}) dt \right. \\ &\left. + (t_s^m - T_c)(1 - e^{-\lambda^m T_c}) \right\} dF(t_s^m).\end{aligned} \tag{18}$$

4.2 Proactive Invalidation With Removing

For proactive invalidation with removing, not only *eviction* but also *invalidity* makes content copies nonexistent in the cache, causing the cache to be not always full. With regard to this fact, the cache occupancy size O can be represented by means of two bounds: (i)when cache is full, evidently the first bound is C, (ii)otherwise, only the valid contents can stay in the cache and the second bound can be expressed approximately as $\sum_{m=1}^{M} 1 \cdot P_v^m$. We take the minimum value of both

$$\begin{aligned}O &\approx \min\left\{C, \sum_{m=1}^{M} 1 \cdot P_v^m\right\} \\ &= \min\left\{C, \sum_{m=1}^{M} \frac{1}{\mu_s^m} \int_0^{\infty} \int_0^{t_s^m} (1 - e^{-\lambda^m t}) dt dF(t_s^m)\right\}.\end{aligned} \tag{19}$$

In addition, the time that a content copy moves from the top to the bottom and finally to the outside of the cache without any hits taking place (*eviction* event) cannot be characterized by T_c due to the *invalidity* event. Let T_c' denote the new characteristic time.

Hence, by substituting T_c' for T_c in (17), previous analytical expressions obtained for reactive invalidation can be rewritten as follows:

$$\begin{aligned}P_h^m = &\frac{1}{\mu_s^m} \int_0^{T_c'} \int_0^{t_s^m} (1 - e^{-\lambda^m t}) dt dF(t_s^m) \\ &+ \frac{1}{\mu_s^m} \int_{T_c'}^{\infty} \left\{ \int_0^{T_c'} (1 - e^{-\lambda^m t}) dt \right. \\ &\left. + (t_s^m - T_c')(1 - e^{-\lambda^m T_c'}) \right\} dF(t_s^m),\end{aligned} \tag{20}$$

and the value of T_c' can be derived by calculating the unique root of the following equation:

$$O = \sum_{m=1}^{M} P_h^m. \tag{21}$$

4.3 Proactive Invalidation With Renewing

For proactive invalidation with renewing, when the server updates a content, it will push a new content copy to the cache to renew rather than to remove the stale one. Hence the *invalidity* event has no influence on the cache hit behavior. In other words, as long as a content copy exists, the request for it will yield a cache hit. According to (2), we have

$$P_h^m = 1 - e^{-\lambda^m T_c}. \tag{22}$$

4.4 Proactive Invalidation With Optional Renewing

Different from proactive invalidation with renewing, proactive invalidation with optional renewing only updates the top M_p popular contents. For unpopular ones, it will remove the stale copies of them from the cache. Let us denote the hit probabilities of popular contents and unpopular contents as $P_{h\ \mathrm{pop}}^m$ and $P_{h\ \mathrm{unpop}}^m$ respectively. Thus, we have

$$P_h^m = \begin{cases} P_{h\ \mathrm{pop}}^m, m \le M_p; \\ P_{h\ \mathrm{unpop}}^m, otherwise. \end{cases} \tag{23}$$

where

$$P_{h\ \mathrm{pop}}^m = 1 - e^{-\lambda^m T_c''}, \tag{24}$$

and

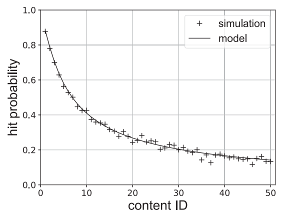
$$\begin{aligned}P_{h\ \mathrm{unpop}}^m = \quad &\tfrac{1}{\mu_s^m} \textstyle\int_0^{T_c''} \int_0^{t_s^m} (1 - e^{-\lambda^m t}) dt dF(t_s^m) \\ &+ \tfrac{1}{\mu_s^m} \textstyle\int_{T_c''}^{\infty} \left\{ \int_0^{T_c''} (1 - e^{-\lambda^m t}) dt \right. \\ &\left. + (t_s^m - T_c'')(1 - e^{-\lambda^m T_c''}) \right\} dF(t_s^m).\end{aligned} \tag{25}$$

The value of new characteristic time T_c'' can also be obtained by calculating the fixed point equation as follows:

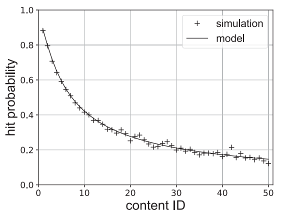
$$O' = \sum_{m=1}^{M} P_h^m = \sum_{m=1}^{M_p} P_{h\ \mathrm{pop}}^m + \sum_{m=M_p+1}^{M} P_{h\ \mathrm{unpop}}^m, \tag{26}$$

where the cache occupancy size O' can be expressed as

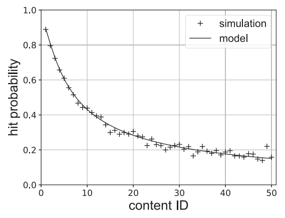
$$O' \approx \begin{cases} \min\{C, M_p + \sum_{m=M_p+1}^{M} P_v^m\} & , M_p < C; \\ C & , otherwise. \end{cases} \tag{27}$$



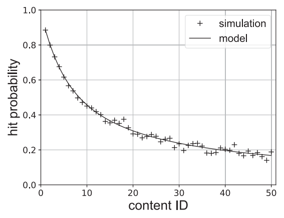
(a) Reactive invalidation, $T_s^m = 10$, $\delta = 0.83\%$.



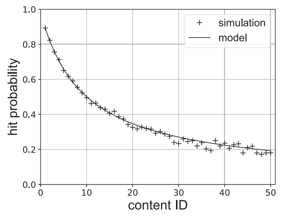
(b) Reactive invalidation, $T_s^m \sim N[0, 20]$, $\delta = 1.03\%$.



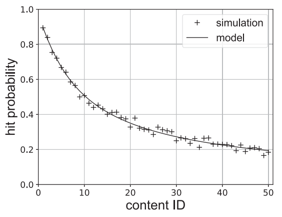
(c) Reactive invalidation, $T_s^m \sim E(0.1)$, $\delta = 0.77\%$.



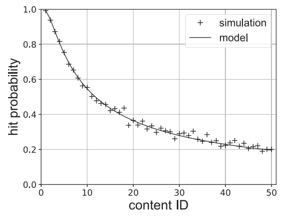
(d) Proactive invalidation with removing, $T_s^m = 10$, $\delta = 0.24\%$.



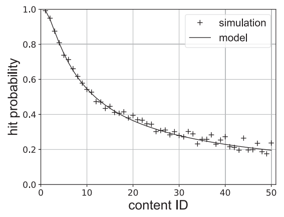
(e) Proactive invalidation with removing, $T_s^m \sim N[0, 20]$, $\delta = 0.62\%$.



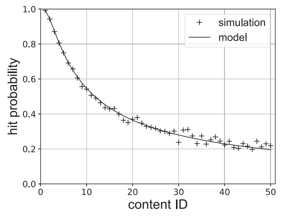
(f) Proactive invalidation with removing, $T_s^m \sim E(0.1)$, $\delta = 0.09\%$.



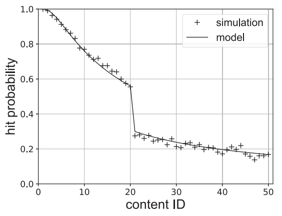
(g) Proactive invalidation with renewing, $T_s^m = 10$, $\delta = 0.49\%$.



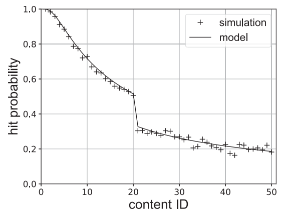
(h) Proactive invalidation with renewing, $T_s^m \sim N[0, 20]$, $\delta = 1.14\%$.



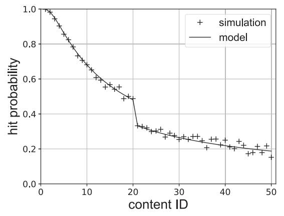
(i) Proactive invalidation with renewing, $T_s^m \sim E(0.1)$, $\delta = 0.39\%$.



(j) Proactive invalidation with optional renewing, $M_p = 20$, $T_s^m = 10$, $\delta = 0.43\%$.



(k) Proactive invalidation with optional renewing, $M_p = 20$, $T_s^m \sim N[0, 20]$, $\delta = 1.15\%$.



(l) Proactive invalidation with optional renewing, $M_p = 20$, $T_s^m \sim E(0.1)$, $\delta = 0.08\%$.

Fig. 5. Cache hit probability for individual contents.

4.5 Server Load

For reactive invalidation and proactive invalidation with removing, the server load is equal to the miss rate of the cache. Then we have

$$L = \sum_{m=1}^{M} \lambda^m (1 - P_h^m) = \lambda(1 - P_h). \tag{28}$$

For proactive invalidation with renewing, the server not only responds to the miss stream, but also actively pushes content copies to update the stale ones in the cache. Hence the server load can be separated into two parts: direct load and push load. Obviously, the calculation of direct load is the same as (28). With regard to push load, the server pushes a new copy if and only if there is a stale one existing in the cache. For each update point, the probability of the content m's copy existing in the cache is $1 - e^{-\lambda^m T_c}$ based on (2) and the update rate is $1/\mu_s^m$. Then we have

$$L = \lambda(1 - P_h) + \sum_{m=1}^{M} \frac{1}{\mu_s^m}(1 - e^{-\lambda^m T_c}). \tag{29}$$

For proactive invalidation with optional renewing, since the server only updates the first M_p contents, the total server load can be written as

$$L = \lambda(1 - P_h) + \sum_{m=1}^{M_p} \frac{1}{\mu_s^m}(1 - e^{-\lambda^m T_c''}). \tag{30}$$

5 Performance Evaluation

The aim of this section is threefold. First, we validate the analytical expressions obtained previously against simulations to evaluate the accuracy of our model, and also we compare our model with Detti model on reactive invalidation and proactive invalidation with removing. Second, we assess the impact of the cache invalidation on the LRU cache. Third, we compare the four different schemes of cache invalidation under expiration time, cache capacity as well as the number of actively renewed contents respectively and wish to get some common insights on the parameter settings.

In order to implement the experiment, we developed a simulation platform based on Python. A user simulator, a cache simulator as well as a server simulator are created in our platform. The user simulator requests for contents according to the Zipf model. If the requested content is in the cache and not outdated, a cache hit event will be recorded. Otherwise, a cache miss will be recorded. An LRU stack is installed on the cache simulator that is used to respond to the user requests and receive the invalidation messages from the server. Besides, the server simulator is in charge of maintaining the content update information and sending invalidation messages actively or reactively according to different invalidation schemes.

Except where otherwise stated, we consider a cache capacity with $C = 100$, where requests arrive for 5000 contents following a Poisson distribution with mean rate $\lambda =$

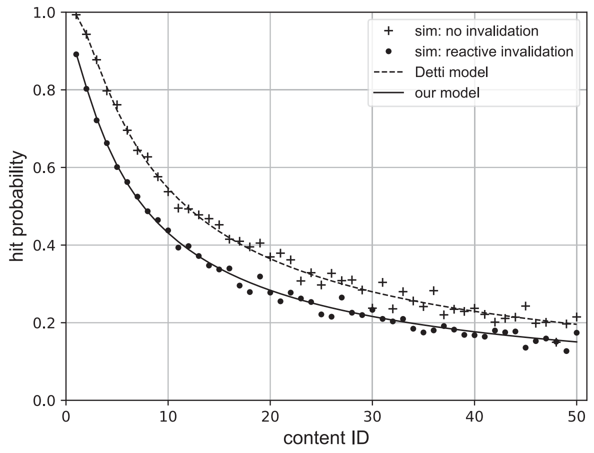


Fig. 6. Comparison between Detti model and our model on reactive invalidation.

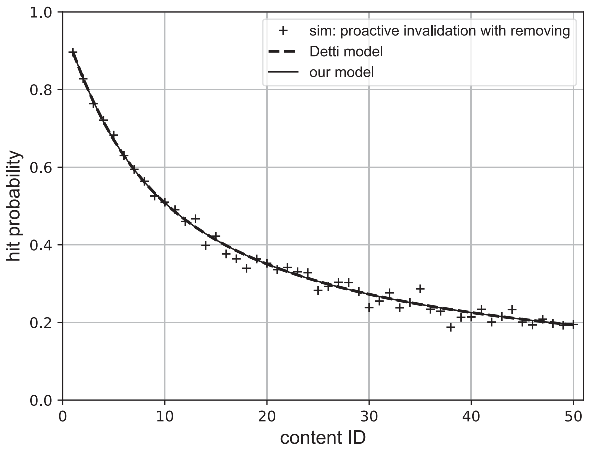


Fig. 7. Comparison between Detti model and our model on proactive invalidation with removing.

$20\ \mathrm{req/s}$. The request probability is modeled by a Zipf distribution with exponent $z = 0.8$. In addition, for proactive invalidation with optional renewing, we set $M_p = 20$. Without loss of generality, it is supposed that the higher the request probability of the content, the lower the content ID.

5.1 Cache Hit Probability

For measuring the accuracy of our model, we denote the hit probability error as

$$\delta = \frac{|P_{h_{\mathrm{model}}} - P_{h_{\mathrm{sim}}}|}{P_{h_{\mathrm{sim}}}}, \tag{31}$$

where $P_{h_{\mathrm{model}}}$ and $P_{h_{\mathrm{sim}}}$ are the mean hit probabilities of the model and simulation respectively.

Fig. 5 characterizes the hit probability of individual contents with four schemes of cache invalidation. Only the first 50 contents are shown to avoid cluttering the figure. For comprehensive assessments, we set up three distributions of expiration time T_s^m which are constant (Figs. 5a, 5d, 5g and 5j), uniformly distributed over [0,20] (Figs. 5b, 5e, 5h, and 5k) and exponentially distributed with mean rate $0.1\mathrm{req/s}$ (Figs. 5c, 5f, 5i, and 5l). In particular, Figs. 5j, 5k, and 5l show an apparent piecewise point at ID $= 20$ for reasons of only the top 20 contents being renewed. As revealed in above figures, the analytical curves match the simulation results surprisingly within the maximum error of less than 1.2 percent, which illustrates the extremely high accuracy of our model.

Given space limitations, we only consider T_s^m with a exponential distribution with mean rate $1/\mu_s^m$ in the following part of this paper.

Fig. 6 shows the comparison between Detti model and our model on reactive invalidation. The results of the simulations of reactive invalidation and no invalidation are also presented in Fig. 6. It is obvious to see that the Detti model for reactive invalidation, in practice, is to describe the no invalidation where the requests for outdated contents stored in the cache are also counted for a cache hit. The error of Detti model between the simulation of no invalidation is 0.34 percent, but it is 22.11 percent between the simulation of reactive invalidation. With this just the opposite is, the error of our model between the simulation of reactive invalidation is only 0.24 percent.

Fig. 7 shows the comparison between Detti model and our model on proactive invalidation. The curves of Detti model and our model are almost coincided and make a quite good agreement on fitting the hit ratio of the simulation of proactive invalidation. In Figs. 6 and 7, the mean expiration time is all set to 10s.

Fig. 8 shows the impact of different mean expiration time μ_s^m on the hit probability P_h. Since the *invalidity* event makes no difference to the cache hit behavior in the proactive invalidation with renewing, the curve of which can be regarded as a baseline that is equivalent to the hit probability without consideration of content expiration. From Fig. 8, we can see that the hit probabilities of reactive invalidation and proactive invalidation with removing increase gradually and approach the baseline while the mean expiration time increasing. The smaller the mean expiration time, the greater the impact of it on the hit probability. Additionally, it is noticed that proactive invalidation with removing has a higher hit probability than reactive invalidation, because the former leaves more space for caching by removing stale content copies from the cache.

The hit probability of proactive invalidation with optional renewing in Fig. 8 is distinctive. The curve of it increases when μ_s^m is less than $4\mathrm{s}$ and then decreases and approaches the baseline from above. To explain this phenomenon, we recall the definition of the average hit probability P_h first. According to (15), highly popular contents contribute a lot to P_h. Therefore, when μ_s^m is small, there is enough space to store the popular contents in the cache, making the hit probability of them close to 1. With the increase of μ_s^m (but still very small), the hit probability of unpopular contents grows, thus causing the overall hit probability to be higher. When μ_s^m is further larger, unpopular contents will crowd out popular ones in the cache, which lessens the hit popularity of the latter and then leads to a reduction in the overall hit probability. Also be noted that, when the cache is in a critical state of saturation or unsaturation (corresponding to the interval [4,8] in Fig. 8), the approximate Equation (27) provides a slight error of less than 4.28 percent.

Fig. 9 shows the impact of different cache capacities C on the hit probability P_h. As the cache capacity increases, the hit probabilities of the four invalidation schemes grow. However, due to the impact of content expiration, (i) the curves of

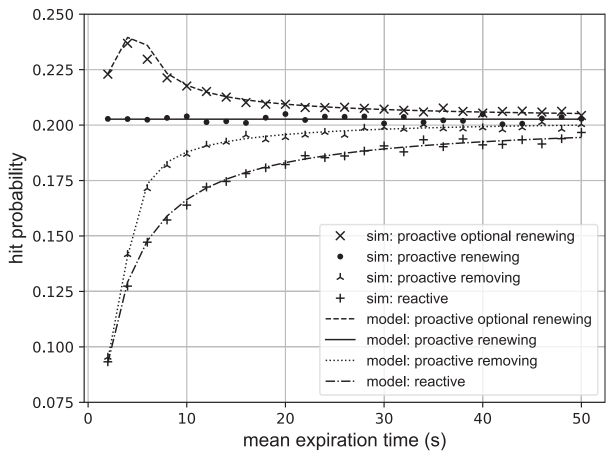


Fig. 8. Hit probability versus mean expiration time, by fixing cache capacity to 100.

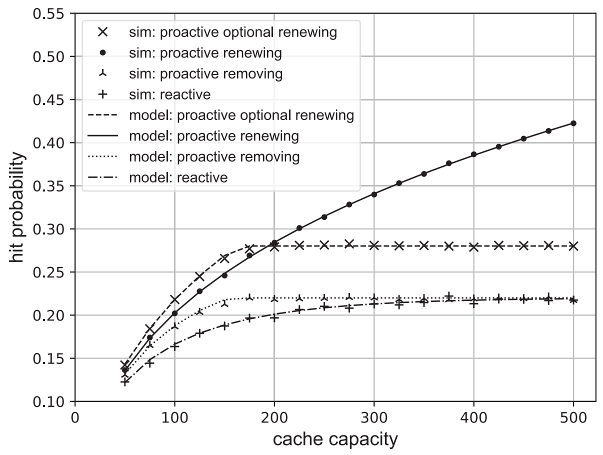


Fig. 9. Hit probability versus cache capacity, by fixing mean expiration time to 10s.

reactive invalidation and proactive invalidation with removing steadily flatten out and tend to merge to the validity probability, (ii) and the curve of proactive invalidation with optional remains basically invariable, while the cache capacity further increases. Since the top 20 popular contents are updated actively and the contribution of hit probability from them is more remarkable, the proactive invalidation with optional renewing has a higher hit probability even than the baseline when the cache capacity is small.

5.2 Server Load

Considering that the requests for stale contents impose additional traffic burdens on the server, we compare the impact of the four invalidation schemes on the server load in this section.

Fig. 10 shows the impact of different expiration time μ_s^m on the server load. Owing to indiscriminately updating, the proactive invalidation with renewing produces much more load for the server. Similarly, the proactive invalidation with optional renewing also produces the extra update traffic of the top 20 popular contents, hence resulting in apparently more load for server compared to the reactive invalidation and proactive invalidation with removing when μ_s^m is small. However, the extra load of proactive invalidation with optional renewing is much lower than that of the proactive

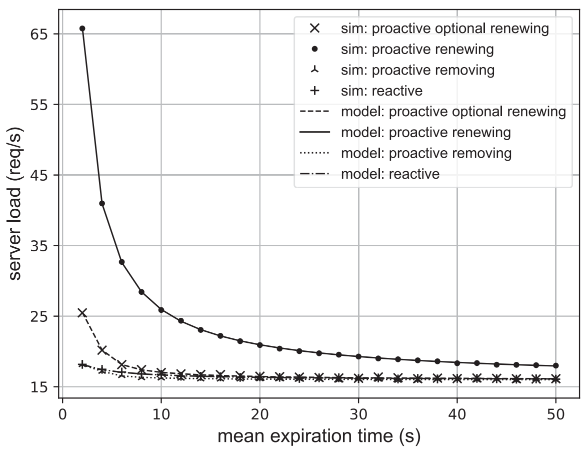


Fig. 10. Server load versus mean expiration time, by fixing cache capacity to 100.

invalidation with renewing. Besides, associating Fig. 10 with Fig. 8, it is seen that the proactive invalidation with optional renewing can achieve a great improvement on hit ratio without bringing too great extra load when the server updates the contents at a high frequency. As μ_s^m increases, the server updates contents at a slower rate and generates lower update load, which brings the server load of the four schemes of cache invalidation basically consistent.

Fig. 11 shows the impact of different cache capacities C on the server load. The curve of the proactive invalidation with renewing grows nearly linearly with the increase of C, due to the server updating all contents stored in the cache. By contrast, the other three schemes of cache invalidation produce lower server load steadily, among which the proactive invalidation with optional renewing produces slightly more (less than 4.37 percent in Fig. 11).

5.3 The Setting of M_p in Proactive Invalidation With Optional Renewing

By comparing Figs. 9 and 11, we observe that the proactive invalidation with optional renewing generates almost identical load for the server with the reactive invalidation and proactive invalidation with removing. However, the hit probability of the former is up to 28.77 percent higher than that of the latter. Additionally, since the small cache capacity limits the amount of the cached contents, most of the contents in the cache are the popular ones. Based on this fact, proactive optional renewing is able to outperform the proactive renewing in both hit rate and server load when the cache capacity is small. Concretely speaking, (i) in case of the small cache capacity of Fig. 9, for the proactive optional renewing, the positive updating mechanism allows popular contents to stay in the cache longer and the positive eviting mechanism makes unpopular contents stay in the cache shorter. Under the co-action of these two mechanisms, the cache with small capacity can store more popular contents which contribute significantly more to the total cache hit probability. Thus, the hit probability of the proactive optional renewing is higher than that of the proactive renewing. (ii) In case of the small cache capacity of Fig. 11, as soon as the content (whether it is popular or not) in the cache is updated, the server in the proactive renewing scheme will push a new copy. However, in the proactive

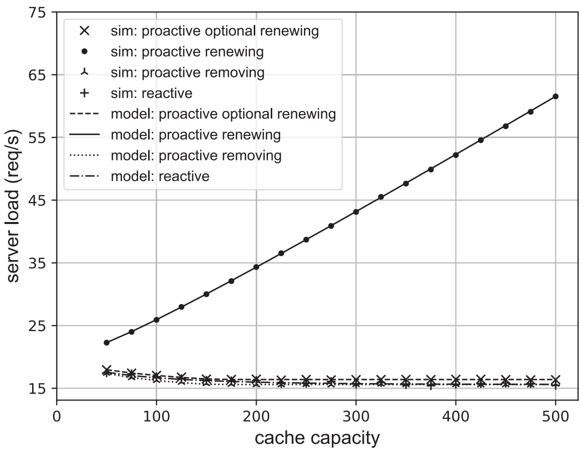


Fig. 11. Server load versus cache capacity, by fixing mean expiration time to 10s.

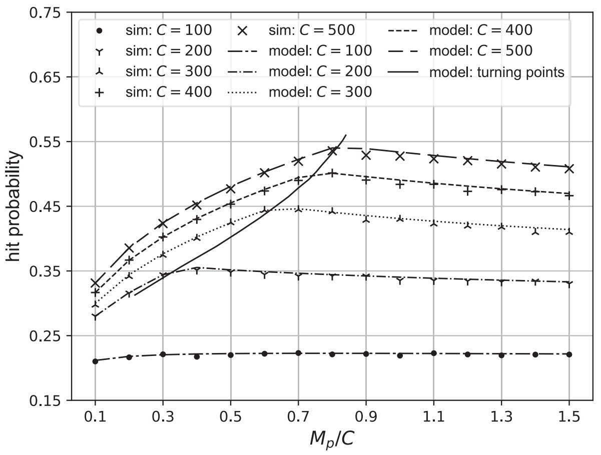


Fig. 12. Hit probability versus M_p/C for proactive invalidation with optional renewing, by fixing mean expiration time to 10s.

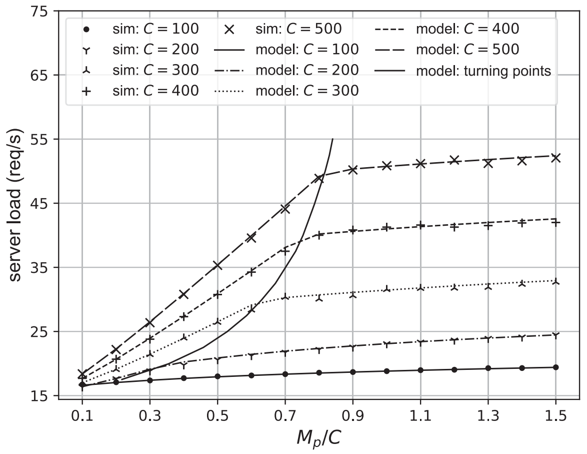


Fig. 13. Server load versus M_p/C for proactive invalidation with optional renewing, by fixing mean expiration time to 10s.

optional renewing scheme, the pushing event only occurs when the popular content in the cache is updated. Thus, the server load of the proactive optional renewing is lower than that of the proactive renewing. In the following section, we will pay particular attention to the performance of the proactive invalidation with optional renewing and derive some principles on the parameter setting.

Since the key to the proactive invalidation with optional renewing is to update the top M_p popular contents and the cache capacity also has an impact on the setting of M_p, we set different M_p/C and record the corresponding hit probability and server load as shown in Figs. 12 and 13 respectively. It is noticed that with M_p growing, the hit probability increases first and then decreases, while the server load increases and basically flattens out. Therefore, an appropriate M_p/C should be selected to balance the hit probability and server load of which the proactive invalidation with optional renewing can achieve better performance.

Furthermore, from Figs. 12 and 13, we can see an apparent turning point in each curve when $C \geq 200$. To figure out this point, let us recall the curve of proactive invalidation with optional renewing in Fig. 8, where when the cache is in a critical state of saturation or unsaturation, the hit probability reaches maximum. Thus we can set M_{p0} to keep the cache in this state and obtain these turning points.

According to (27), this state can be expressed approximately as

$$C \approx M_{p0} + \sum_{m=M_{p0}+1}^{M} P_v^m. \tag{32}$$

Hence, M_{p0} can be derived by calculating the positive integer root of the above equation, if and only if $C > \sum_{m=1}^{M} P_v^m$. Then we redraw two curves of proactive invalidation with optional renewing with different cache capacities in Figs. 12 and 13 respectively, where the new curves (solid curves) basically pass through the turning point of each original curve.

The customizable parameter setting on M_p in the proactive invalidation with option renewing greatly enhances the flexible application of this scheme. By setting M_p to different values, we can obtain different combinations of hit probability and server load to meet the demands of various cache systems. Figs. 12 and 13 demonstrate that: (i) the increase of M_p causes both hit probability and server load to rise rapidly, and then causes the hit probability to fall slowly and the server load to rise slowly; (ii) the variation magnitude of the hit probability becomes larger than that of the server load as the cache capacity growing. Further, based on the above two points respectively, it is concluded that (i) limiting M_p to a smaller value could achieve a lower server load, but at the cost of lower hit probability; (ii) the cost of increasing the server load by raising the value of M_p outweighs the benefit of increasing the cache hit probability, and the gap between the cost and benefit becomes larger with the increase of cache capacity. Therefore, we could gain some principles on the setting of M_p: (i) when the server has sufficient load capacity, choosing M_p close to M_{p0} can improve the hit probability greatly, (ii) otherwise, choosing a relatively small M_p can maintain the server load at a low level but still obtains a fairly high hit probability.

5.4 The Performance Evaluation in Realistic Traffic

In order to evaluate four invalidation schemes performance, we apply our model to realistic traffic data that is obtained from the open APIs provided by Wikipedia.[1] It is important

1. Wikipedia's open interface: https://wikimedia.org/api/rest_v1

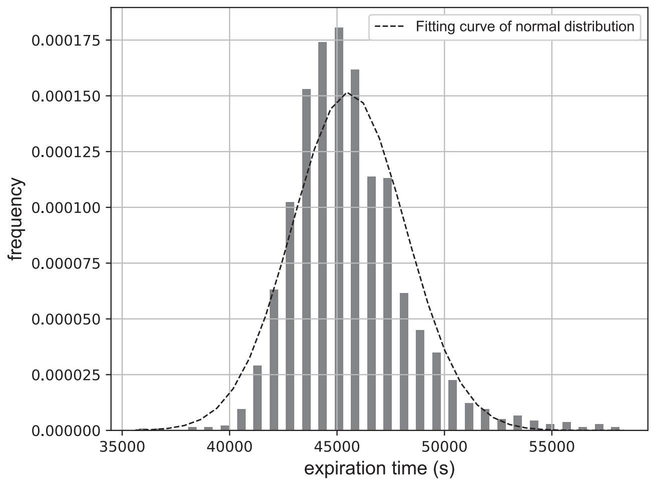


Fig. 14. The distribution of content expiration time counted from Wikipedia from January 1, 2015 to December 31, 2019, which is fitted with a norm distribution whose shape parameters are $\mu = 45547$ and $\sigma = 2630$.

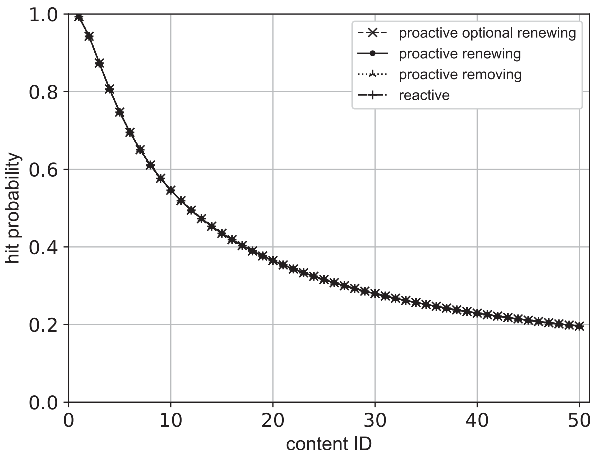


Fig. 15. The hit probabilities of four invalidation schemes for individual contents under the norm distribution of expiration time with $\mu = 45547$ and $\sigma = 2630$.

to note that the APIs only offer the number of page edited times and corresponding date, and we use the interval between two consecutive editing events as the content expiration time. In the following part, first, we analyze the real distribution of content expiration time. Second, we compare four invalidation schemes with real distribution of content expiration time. Third, we reduce the mean expiration time to reflect the effect of invalidation on cache hit probability.

Fig. 14 shows the real distribution of content expiration time in Wikipedia. The data ranges from January 1, 2015 to December 31, 2019. Since the smallest time interval is day in the data, we use the ratio of the number of all page edits to the number of edited pages for the day to approximately represent the single page edits per day, and then we use the ratio of $24 * 3600(s)$ to the single page edited times per day to present the expiration time of the single page. The counting result approximately follows a norm distribution with $\mu = 45547$ and $\sigma = 2630$.

Similarly, to avoid cluttering the figure, only the hit probabilities of the top 50 contents are marked in Fig. 15. Since the hit probability of proactive invalidation with renewing is not affected by the expiration time, the curve of it can be regarded as a curve without invalidation. It is clearly to see that the curves of four invalidation schemes are almost coincide. That is because the mean content expiration time is $45547s$ which is large enough to eliminate the impact of invalidation on cache hit probability. The content is evicted from the cache due to the cache capacity limitation far before it reaches the expiration time.

Fig. 16 shows the impact of different mean expiration time on the hit probability of four invalidation schemes. It has been seen that the large mean expiration time has scarcely influence upon hit probability in Fig. 15. Thus, we decrease the mean expiration time, μ, to improve the influence of invalidation. Besides, in order to maintain the shape of the norm distribution, the other parameter, σ, is scaled down. The distributions of expiration time in Fig. 16 are $N(45547, 2630^2)$, $N(4554, 262.9^2)$, $N(455, 26.3^2)$ and $N(45, 2.6^2)$ in descending order of μ. As shown in Fig. 16, with the reduction of mean expiration time, the curve of

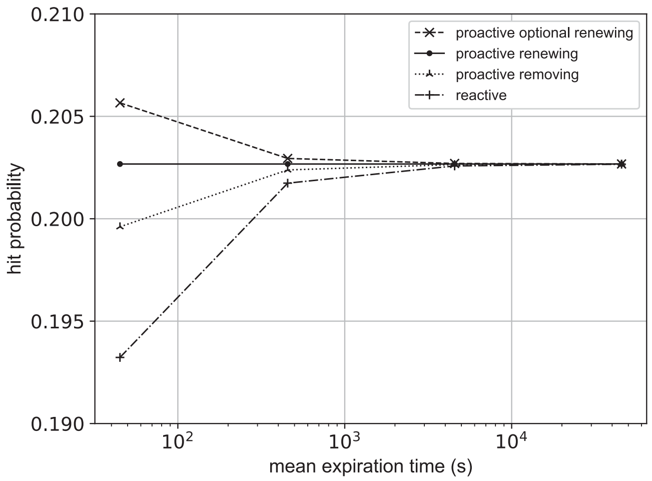
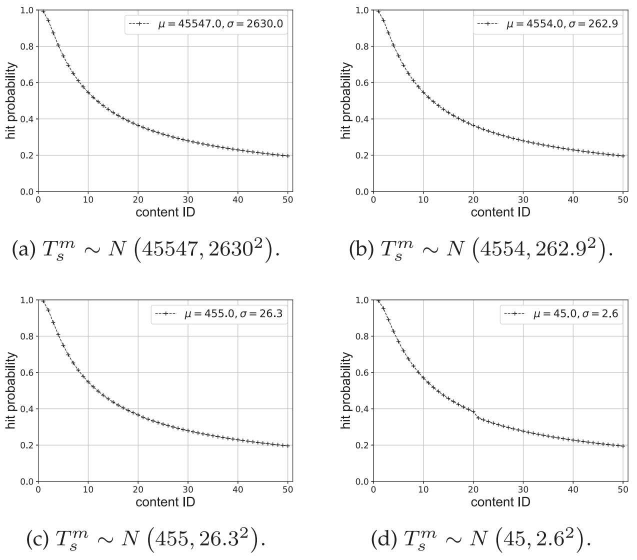


Fig. 16. The hit probability of four invalidation schemes versus mean expiration time. The values of mean expiration time are $45s$, $455s$, $4554s$ and $45547s$ respectively.

four invalidation schemes gradually diverge, which means that the impact of invalidation on hit probability is gradually strengthened. The hit probabilities of the four invalidation schemes are proactive optional renewing, proactive renewing, proactive removing and reactive from the largest to the smallest, which is also consistent with the order in Fig. 8.

Fig. 17 shows the hit probability of proactive invalidation with optional renewing for individual contents under $N(45547, 2630^2)$, $N(4554, 262.9^2)$, $N(455, 26.3^2)$ and $N(45, 2.6^2)$ in descending order of μ respectively. For proactive invalidation with optional renewing, the hit probabilities of popular contents and unpopular contents are different. However, this difference is not significant in the curve due to the large mean expiration time as shown in Figs. 5a, 5b, and 5c. Only when the mean expiration time is relatively small as shown in Fig. 5d, the difference become obvious and an apparent piecewise point appears at ID = 20. The advantages of proactive invalidation with optional renewing over other three schemes also become notable as shown in Fig. 16 at $\mu = 45$.



(a) $T_s^m \sim N\left(45547, 2630^2\right)$. (b) $T_s^m \sim N\left(4554, 262.9^2\right)$.

(c) $T_s^m \sim N\left(455, 26.3^2\right)$. (d) $T_s^m \sim N\left(45, 2.6^2\right)$.

Fig. 17. The hit probability of proactive invalidation with optional renewing for individual contents under the norm distribution of expiration time. Only the top 20 contents are positively renewed.

5.5 The Summary of Simulation Results and Insights

In this section, we measure the performance of the proposed model, draw comparisons with an existing model, assess the invalidation impact on the LRU cache and compare four different schemes of cache invalidation. Through the above simulations, some important results and insights can be summarized as follows:

- Our model can achieve a fairly high accuracy on predicting the hit probability and server load of four cache invalidation schemes.
- Our model is more accurate than the currently proposed model in calculating the hit probability of the reactive invalidation.
- Due to the mechanism of positively updating popular contents, the invalidation with proactive optional renewing has a better performance against other invalidation schemes when the cache capacity is tight and the content expires frequently.
- When the cache is in the critical state of saturation and unsaturation, the hit probability of the invalidation with proactive optional renewing can reach maximum. Surprisingly, this cache state can be realized by setting M_p in accordance with our model.
- In general, the proactive invalidation with proactive optional renewing can be prioritized when the system requires strong cache consistency. Particularly, by setting M_p to reasonable values in different scenes as we mentioned in Section 5.3, this invalidation scheme can achieve better performance on balancing the hit probability and server load.
- In the simulation with the real distribution of content expiration time, the effect of invalidation on cache performance is obvious only when the mean expiration time is small.
- In some networks with real-time features (e.g., IoT networks), the content expiration time is small. If the server performance is high and the network bandwidth is abundant, applying proactive invalidation with optional renewing is a good choice. Otherwise, this scheme can also be used, because the server load can be reduced by decreasing the number of actively renewed contents at the expense of a little hit probability.

6 Conclusion

In this paper, we address the problem of modeling four schemes of cache invalidation with LRU replacement. By applying conditional probability to characterize the interactions between existence and validity, we derive analytical expressions for cache hit probability and server load under arbitrary frequency distribution. The simulation results demonstrate that our model is able to achieve more extensibility and exceedingly high accuracy than the Detti model. Also, we compare the performance of four different invalidation schemes, among which the proactive invalidation with optional renewing has better scalability and, further, can obtain the performance balance by reasonable parameter settings.

Though the model of invalidation presented here is for LRU caches, we believe it can be adapted to other caches with which the characteristic time is able to be associated. In addition, the insights gained through a single cache also can be expanded to guide the design of invalidation schemes in cache networks. Furthermore, how the time-varying content popularity, future traffic (e.g., IoT) as well as the distribution of contents' copies affect the model of cache invalidation are also attractive topics. We would keep on investigating these issues in the future.

Acknowledgments

This work was supported in part by the National Key R&D Program of China under Grant 2020YFA0711400, in part by the Key R&D Program of Anhui Province in 2020 under Grant 202004a05020078 and in part by the CETC Joint Advanced Reasearch Foundation under Grant 6141B08080101. We also would like to acknowledge the supports from the Institute of Artificial Intelligence, Hefei Comprehensive National Science Center and the Institute of Advanced Technology, University of Science and Technology of China.

References

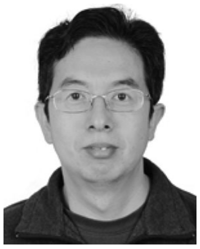
[1] S. Podlipnig and L. Böszörmenyi, "A survey of web cache replacement strategies," *ACM Comput. Surv. (CSUR)*, vol. 35, no. 4, pp. 374–398, Dec. 2003.
[2] S. Lee, I. Yeom, and D. Kim, "T-caching: Enhancing feasibility of in-network caching in ICN," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 7, pp. 1486–1498, Jul. 2020.
[3] J. Jung, A. W. Berger, and H. Balakrishnan, "Modeling TTL-based internet caches," in *Proc. IEEE 22nd Annu. Joint Conf. IEEE Comput. Commun. Soc.*, 2003, pp. 417–426.
[4] J. Gwertzman and M. I. Seltzer, "World wide web cache consistency," in *Proc. USENIX Annu. Techn. Conf.*, 1996, pp. 141–152.
[5] O. Bahat and A. M. Makowski, "Measuring consistency in TTL-based caches," *Perform. Eval.*, vol. 62, no. 1–4, pp. 439–455, 2005.
[6] N. C. Fofack, P. Nain, G. Neglia, and D. Towsley, "Analysis of TTL-based cache networks," in *Proc. 6th Int. ICST Conf. Perform. Eval. Methodol. Tools*, 2012, pp. 1–10.
[7] C. Gray and D. Cheriton, "Leases: An efficient fault-tolerant mechanism for distributed file cache consistency," *ACM SIGOPS Operating Syst. Rev.*, vol. 23, no. 5, pp 202–210, 1989.

[8] B. Krishnamurthy and C. E. Wills, “Piggyback server invalidation for proxy cache coherency,” *Comput. Netw. ISDN Syst.*, vol. 30, no. 1–7, pp. 185–193, 1998.
[9] G. Cao, “A scalable low-latency cache invalidation strategy for mobile environments,” *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 5, pp. 1251–1265, 2003.
[10] J. Jing, A. Elmagarmid, A. S. Helal, and R. Alonso, “Bit-sequences: An adaptive cache invalidation method in mobile client/server environments,” *Mobile Netw. Appl.*, vol. 2, no. 2, pp. 115–127, 1997.
[11] A. Detti, L. Bracciale, P. Loreti, and N. B. Melazzi, “Modeling LRU cache with invalidation,” *Comput. Netw.*, vol. 134, pp. 55–65, 2018.
[12] M. Rabinovich and O. Spatscheck, *Web Caching and Replication*. Boston, MA, USA: Addison-Wesley, 2002.
[13] R. Fielding *et al.*, “Hypertext transfer protocol–http/1.1,” 1999. [Online]. Available: https://www.w3.org/Protocols/rfc2616/rfc2616.html
[14] A. Iyengar, E. Nahum, A. Shaikh, and R. Tewari, “Web caching, consistency, and content distribution,” *The Practical Handbook of Internet Computing*, Boca Raton, FL, USA: CRC Press, 2004.
[15] A. Iyengar, E. Nahum, A. Shaikh, and R. Tewari, “Enhancing web performance,” in *Proc. IFIP World Comput. Congr.*, 2002, pp. 95–126.
[16] K.-L. Tian, J. Cai, and B. C. Ooi, “An evaluation of cache invalidation strategies in wireless environments,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 12, no. 8, pp. 789–807, Aug. 2001.
[17] E. J. O’neil, P. E. O’neil, and G. Weikum, “The LRU-K page replacement algorithm for database disk buffering,” *ACM SIGMOD Rec.*, vol. 22, no. 2, pp. 297–306, 1993.
[18] D. Lee *et al.*, “LRFU: A spectrum of policies that subsumes the least recently used and least frequently used policies,” *IEEE Trans. Comput.*, vol. 50, no. 12, pp. 1352–1361, Dec. 2001.
[19] W. F. King, “Analysis of paging algorithms,” in *Proc. IFIP Congr.*, 1971, pp. 485–490.
[20] P. Flajolet, D. Gardy, and L. Thimonier, “Birthday paradox, coupon collectors, caching algorithms and self-organizing search,” *Discrete Appl. Math.*, vol. 39, no. 3, pp. 207–229, 1992.
[21] A. Dan and D. Towsley, “An approximate analysis of the LRU and FIFO buffer replacement schemes,” *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 18, no. 1, pp. 143–152, Apr. 1990.
[22] E. J. Rosensweig, J. Kurose, and D. Towsley, “Approximate models for general cache networks,” in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[23] H. Che, Y. Tung, and Z. Wang, “Hierarchical Web caching systems: Modeling, design and experimental results,” *IEEE J. Sel. Areas Commun.*, vol. 20, no. 7, pp. 1305–1314, Sep. 2002.
[24] C. Fricker, P. Robert, and J. Roberts, “A versatile and accurate approximation for LRU cache performance,” in *Proc. 24th Int. Teletraffic Congr.*, 2012, pp. 1–8.
[25] V. Martina, M. Garetto, and E. Leonardi, “A unified approach to the performance analysis of caching systems,” in *Proc. IEEE Conf. Comput. Commun.*, 2014, pp. 2040–2048.
[26] X. Tang, H. Chi, and S. T. Chanson, “Optimal replica placement under TTL-based consistency,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 3, pp. 351–363, Mar. 2007.
[27] M. Amadeo, C. Campolo, G. Ruggeri, G. Lia, and A. Molinaro, “Caching transient contents in vehicular named data networking: A performance analysis,” *Sensors*, vol. 20, no. 7, 2020, Art. no. 1985.
[28] S. Vural, N. Wang, P. Navaratnam, and R. Tafazolli, “Caching transient data in internet content routers,” *IEEE/ACM Trans. Netw.*, vol. 25, no. 2, pp. 1048–1061, Apr. 2017.
[29] E. Leonardi and G. L. Torrisi, “Least recently used caches under the shot noise model,” in *Proc. IEEE Conf. Comput. Commun.*, 2015, pp. 2281–2289.
[30] S. M. Ross *et al.*, *Stochastic Processes*. New York, NY, USA: Wiley, 1996.

Quan Zheng received the BS degree in production process automation from the Dalian University of Technology, Dalian, China, in 1992, and the MS degree in automatic control theory and application and the PhD degree in computer software and theory from the University of Science and Technology of China, Hefei, China, in 1995 and 2003, respectively. He is currently an associate professor with the Department of Automation and the deputy director of Laboratory for Future Networks. His research interests include video semantic retrieval, media content distribution, video quality detection, and future networks.

Tao Yang received the BS degree in 2019 from the Department of Automation, University of Science and Technology of China, Hefei, China, where he is currently working toward the MS degree with the Department of Automation. His current research interests include caching and modeling of NDN.

Yuanzhi Kan received the BS degree in 2015 from the Department of Automation, University of Science and Technology of China, Hefei, China, where he is currently working toward the MS degree with the Department of Automation. His current research interests include caching and modeling of NDN.


Xiaobin Tan received the BS and PhD degrees from the University of Science and Technology of China (USTC), Hefei, China, in 1996 and 2003, respectively. He is currently an associate professor with the School of Information Science and Technology, USTC. His research interests include network performance optimization and information security.

Jian Yang (Senior Member, IEEE) received the BS and PhD degrees from the University of Science and Technology of China (USTC), Hefei, China, in 2001 and 2006, respectively. From 2006 to 2008, he was a postdoctoral scholar with the Department of Electronic Engineering and Information Science, USTC. Since 2008, he has been an associate professor with the Department of Automation, USTC. He is currently a professor with the School of Information Science and Technology, USTC. His research interests include future network, distributed system design, modeling and optimization, and multimedia over wired and wireless and stochastic optimization. He was the recipient of Lu Jia-Xi Young Talent Award from the Chinese Academy of Sciences in 2009.

Xiaofeng Jiang (Member, IEEE) received the BS and PhD degrees from the University of Science and Technology of China (USTC), Hefei, China, in 2008 and 2013, respectively. He is currently an associate research fellow with the School of Information Science and Technology, USTC. His research interests include partially observable arkov decision processes (POMDPs), big data and higher-order tensor analysis, future network performance analysis and modeling, and cognitive wireless and radar counter measures.