IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 29, 2019, accepted January 2, 2020, date of publication January 14, 2020, date of current version January 31, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2966495

Estimating Video Popularity From Past Request Arrival Times in a VoD System

TIANJIAO WANG[1], CHAMIL JAYASUNDARA[2], MOSHE ZUKERMAN[1], (Life Fellow, IEEE), AMPALAVANAPILLAI NIRMALATHAS[2], (Senior Member, IEEE), ELAINE WONG[2], CHATHURIKA RANAWEERA[3], CHANG XING[1], AND BILL MORAN[2], (Life Member, IEEE)

[1]Department of Electrical Engineering, City University of Hong Kong, Hong Kong
[2]Department of Electrical and Electronic Engineering, The University of Melbourne, Melbourne, VIC 3010, Australia
[3]School of Information Technology, Deakin University, Geelong, VIC 3220, Australia

Corresponding author: Moshe Zukerman (moshezu@gmail.com)

This work was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China, under Project 11200417.

ABSTRACT Efficient provision of Video-on-Demand (VoD) services requires that popular videos are stored in a cache close to users. Video popularity (defined by requested count) prediction is, therefore, important for optimal choice of videos to be cached. The popularity of a video depends on many factors and, as a result, changes dynamically with time. Accurate video popularity estimation that can promptly respond to the variations in video popularity then becomes crucial. In this paper, we analyze a method, called *Minimal Inverted Pyramid Distance* (MIPD), to estimate a video popularity measure called the *Inverted Pyramid Distance* (IPD). MIPD requires choice of a parameter, k, representing the number of past requests from each video used to calculate its IPD. We derive, analytically, expressions to determine an optimal value for k, given the requirement on ranking a certain number of videos with specified confidence. In order to assess the prediction efficiency of MIPD, we have compared it by simulations against four other prediction methods: Least Recency Used (LRU), Least Frequency Used (LFU), Least Recently/Frequently Used (LRFU), and Exponential Weighted Moving Average (EWMA). Lacking real data, we have, based on an extensive literature review of real-life VoD system, designed a model of VoD system to provide a realistic simulation of videos with different patterns of popularity variation, using the Zipf (heavy-tailed) distribution of popularity and a non-homogeneous Poisson process for requests. From a large number of simulations, we conclude that the performance of MIPD is, in general, superior to all of the other four methods.

INDEX TERMS Popularity prediction, video-on-demand, pre-placement, request statistic, Zipf distribution, non-homogeneous Poisson process.

I. INTRODUCTION

In recent years, the popularity of Video-on-Demand (VoD) services has increased tremendously. A growing number of people have begun to use on-demand video services because of the unique flexibility of VoD services to provide *what* the users want, *when* they want it. With this rapid increase in popularity, the traffic generated has also dramatically increased. According to the Cisco Visual Networking Index [1], the amount of VoD traffic will nearly double by 2022 and will be equivalent to 10 billion DVDs per month. The sum of IP video will continue to account for 80% to 90% of total IP traffic. Globally, IP video traffic will represent 82% of total traffic by 2022.

The associate editor coordinating the review of this manuscript and approving it for publication was Jenny Mahoney.

Service providers are exploiting several methods to keep up with the ever-increasing demand. One widely considered method to enhance the performance of video distribution networks is to pre-place *popular* objects in strategic locations within the network, based on the anticipated demand [2]–[6]. Given the enormous sizes of video objects and constrained capacities of storage, it is impossible to replicate all videos in all locations. In particular, we can store only a few videos in content storage located towards the edge of the network. Moreover, unlike other objects in web systems, transferring and deletion of video objects generally takes time because of their enormous sizes [7]. The set of items stored in a storage that serves a group of users needs to be *just the right set* taking account of the requirements of that group of users. (We use the terms, objects, items, and video, to refer to any content offered under VoD service, including full-length

videos, drama series and news items.) This is crucial in deciding which video is worth caching and will be requested sufficiently in the future. Effective cache management can be achieved by finding the most popular videos. Here, the popularity of a video is measured by the request count from a given group of users under consideration. Popularity prediction is a process to estimate the number of requests for videos in the future.

The actual popularity of videos can be affected by other factors, such as, ratings, reviews, recommendations, advertising, discounted pricing, etc., which might be hard to quantify [8]–[11]. The study of video popularity estimation is complex and has many directions. One of the main research areas is based on the features (characteristics) of the videos. Trzciński and Rokita [12] use video features (such as visual and social features) to predict the popularity of online videos. By this method, the VoD system can have prior information about the popularity of the videos before they are introduced to the system.

Another crucial research area is an in-depth analysis of the dynamics of the video request arrival process [9]–[11], [13]. In this area, many methods have been proposed to predict popularity based on the request statistics. These methods need to overcome prediction challenges associated with the fact that video popularity is highly dynamic and time-varying and request patterns from users are usually not uniform. To capture those variations, changes in video popularity need to be monitored continuously. Popularity prediction based on video requests has a crucial role in overall video popularity prediction, as this method can quickly adjust to newly available data on video demands.

Our paper focuses on the second area, namely dynamic popularity prediction based on historical request statistics of users. In particular, we provide new analyses and extensive simulation studies based on realistic traffic models to evaluate the performance of a range of methods that address the challenges of this research area. Other aspects of caching are beyond the scope of this paper.

We provide an in-depth performance comparison with four other dynamic popularity estimation methods, namely, Least Recency Used (LRU), Least Frequency Used (LFU), Least Recently/Frequently Used (LRFU), and Exponential Weighted Moving Average (EWMA).

The *Last-k Algorithm* was described in [14] and demonstrated to respond faster than LFU to newly introduced videos. In addition, a formula was derived to approximate the number of videos that should be stored in the cache to achieve a given *hit ratio* (HR) defined as the proportion of the number of requests served by the local server (cache) to the total number of videos requested. The Last-k Algorithm estimates popularity ranking of videos using their most recent request statistics. The parameter k is the number of past request arrivals for each of the videos. Increasing the value of k improves the accuracy of the algorithm at the expense of the response time, computational complexity, and overhead. This suggests the need for a method to optimize the value of k for a given set of requirements, and this was not provided in [14].

The contribution of this paper beyond the contribution of [14] is as follows. Firstly, we provide two methods to optimize k to meet the accuracy requirement through an analysis of the Last-k Algorithm, leading to a new algorithm of popularity estimation that we call the *Minimal Inverted Pyramid Distance* (MIPD) algorithm. MIPD is based on a novel entity called Inverted Pyramid Distance (IPD), which is used to quickly identify popular videos given the arrival times of a recent sequence of video requests. MIPD is achieved by emphasizing the importance of recent requests using a pyramid-like weight. Like its predecessor the Last-k Algorithm, MIPD also uses the parameter k used in the Last-k Algorithm for popularity prediction. Again, in MIPD we have a trade-off between accuracy (confidence level) that improves with increased k, but at a cost of increased overhead and computational complexity. Since IPDs of videos vary frequently, IPD updates are needed. Such updates are time-consuming especially if the value of k is large. MIPD uses an efficient way to update the IPD of each video.

Secondly, based on extensive new simulation results using realistic traffic models, we compare MIPD to four other methods, namely, LRU, LFU, LRFU, and EWMA. These methods will all be described in the next section. We show that MIPD outperforms the other four popularity estimation methods in terms of Hit Ratio (HR) and the accuracy for estimating the rank of videos of given actual popularity in our designed VoD model.

Thirdly, in addition to the straightforward implementation of the MIPD algorithm which is suitable for scenarios where the popularity of the objects is not required to be estimated very frequently. We have also described an optimizing MIPD algorithm for cache replacement, a faster, leaner method, that can handle situations where cache servers are updated whenever a user requests a video not already stored in the cache.

While the MIPD we proposed here can be potentially used in other systems such as web systems, we mainly focus on VoD systems, limiting our discussion and evaluation only to VoD systems. Content management in web cache systems is often considered less challenging as the steadily falling storage costs have enabled distributed web servers (e.g., proxy cache servers) to store an enormous amount of web content [15], [16]. Moreover, it is known that in VoD systems, higher performance can be potentially achieved by storing partial videos (partial caching) instead of full videos [3], [17], [18]. We do not consider such issues of VoD workload here as the focus of our work is on video popularity estimation based on the request arrival process and not based on content analysis. Irrespective of whether we use partial caching or full caching, having an accurate video popularity estimation method is crucial to achieving better performance.

The remainder of the paper is organized as follows. In Section II, we discuss the existing popularity estimation methods and their applications, include the IPD-based

popularity estimation method (Last-k algorithm). Then we describe the MIPD algorithm for video popularity estimation in Section III which is also based on IPD, optimization of k that achieved the required confidence level, and an efficient way for IPD updates. In Section IV, we undertake a deep analysis of real-life VoD system and build a VoD system model. Many simulations based on the VoD model in Section V are used to analyze the influence of parameter k in MIPD algorithm and to compare the performance of MIPD against the other four methods. Concluding remarks are presented in Section VI.

II. RELATED WORK

There are many different research directions in this field and research papers on optimization in the VoD system are found widely in the literature. As discussed in Section I, we focus on popularity estimation in VoD systems based on request statistics of videos. The underlying principles behind most cache replacement algorithms are popularity estimation methods using the request record [3], [17], [19], [20]. That is, in every cache replacement algorithm in general, the objects least likely to be requested are identified using past access patterns so that those objects can be removed from the cache when required. Instead of analyzing the viewing behaviors of individual users, we consider VoD arrival requests generated by many users for the purpose of popularity estimation with the aim to achieve optimal caching.

The *LFU popularity estimation method* and its variants use frequency of requests to identify the least popular object. Consequently, under the LFU cache replacement policy, the item requested the least number of times in the past is removed from the cache. The LFU cache replacement algorithm is often deployed in web cache servers using the frequency of the requests to identify the least popular object, i.e., the object least likely to be requested by the users in future [15], [21].

While the LFU method captures long-term popularity of objects, it responds poorly to changes in user demand as it does not emphasize recent history over earlier reports. Since LFU cannot distinguish between requests that occurred recently with requests that occurred significantly earlier, it can incorrectly identify many ''has-beens'' as popular objects because of their high request count in the past [20], [22]. For example, when the LFU method is used, earlier episode of a drama series that is no longer popular might remain in the cache for a long time.

The *LRU popularity estimation method* and its extensions use the time distance; that is the time from the last request to the current time. Specifically, by using LRU, the object that is requested least recently will be evicted from the cache when needed. While the LRU popularity estimation method responds promptly to changes in object popularity, it does not take frequency of requests into consideration. Consequently, LRU does not capture the long-term popularity of objects. An uneven request arrival pattern might result in less popular objects being identified as popular ones [20], [23]. For example, when LRU is used, the cache might replace all the popular videos with contents that have recent and short-lived popularity.

The LFU popularity estimation method is more effective when the popularity of the videos is stable, whereas LRU is more effective when popularity is rapidly changing. Also, when carrying out the content pre-placement or cache replacement, content providers may have certain preferences over short-term versus long-term popularity, depending on the schedule with which they introduce new videos into the library and the sizes of their servers. For example, if small caches are used to handle errors in demand prediction, then a cache replacement policy such as the LRU is preferred [22]. On the other hand, the LFU policy is shown to work well for proxy caching environments that are aimed to offload the demand in central VoD servers [22], [24]. Both LFU and LRU popularity estimation methods have less flexibility to adjust to varying requirements. Accordingly, a straightforward solution is to implement different algorithms depending on the requirements of different circumstances.

LRFU popularity estimation is a combination of LFU and LRU popularity estimations, and has been demonstrated to achieve better performance than LFU or LRU [21], [25]. More specifically, the LRFU algorithm de-emphasizes the frequencies of old references by using a combined recency and frequency (CRF) value, which quantifies the likelihood that the object will be requested in the near future [25]. When a cache replacement is required, the algorithm replaces the object with the least CRF value, regarded as the least popular object. The CRF value of object b after the k^{th} reference at time t_k, is given by,

$$C_k(b) = \begin{cases} 1 + \left(\frac{1}{2}\right)^{\tau\delta} C_{t_b}(b), & \text{if } b \text{ is referenced at time } k \\ \left(\frac{1}{2}\right)^{\tau\delta} C_{t_b}(b), & \text{otherwise,} \end{cases} \quad (1)$$

where $\delta = t_k - t_b$, and τ is a parameter ranging from 0 to 1 that controls the decay. The parameter t_b is the time of the most recent reference to the object b before the current reference. On the other hand, in the frequency computation in the GDS algorithm, a similar decay function is used to de-emphasize the significance of past accesses. That is, the frequency of object p is defined iteratively for the $(i+1)^{\text{th}}$ reference to this object as,

$$f_{i+1}(p) = f_i(p) \times 2^{-t/T} + 1, \quad (2)$$

where t is the time elapsed since the last reference and T is a constant that controls the rate of decay [26].

The *EWMA popularity estimation method* calculates the average access interval of each video, where the historical information is forgotten exponentially over time. When a request to a specific video arrives, the average request time interval of that video will be updated as

$$T_{interval} \leftarrow (1-p) * T_{interval} + p * (t - t_{last}), \quad (3)$$

where $T_{interval}$ represents the average access interval of a video, t_{last} is the time when that video is requested most

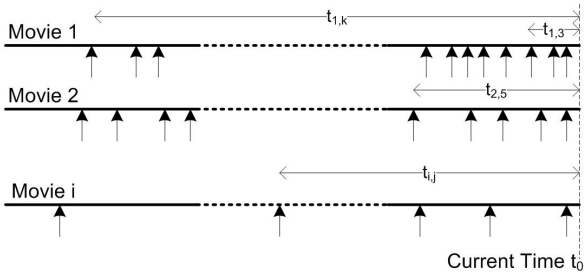


FIGURE 1. Request arrivals at the server.

recently, t is the current time, $(t - t_{last})$ is the latest requested interval, and p is a forgetting rate which ranges between 0 and 1. The newest access interval $(t - t_{last})$ is assigned a weight (probability) $p \in (0, 1]$ in determining the overall average request interval. If p is fixed at $p = 1$, this algorithm becomes LRU [27].

III. MIPD

In this section, we introduce the IPD-based object popularity estimation concept and describe our proposed MIPD method that estimates the popularity of videos using IPD. MIPD is an extension version of Last-k popularity estimation method proposed in [14]. The Last-k method is IPD-based, combining the good features of the LFU and the LRU methods in a very different way than LRFU and EWMA do, whilst avoiding their drawbacks. The Last-k method performs as accurately as the LFU method when the popularity of the videos is stable while responding faster than LFU to newly introduced videos.

In this paper, we undertake a deep analysis of the IPD based popularity estimation method. We show that there is the best value of the parameter k that obtains high accuracy in popularity ranking of video objets using the Last-k ranking via IPD. We also introduce two methods that can be used to find suitable values for k. In addition, we discuss how we can reduce the overhead and memory requirements of MIPD for caching replacement.

A. WHAT IS IPD

In MIPD popularity estimation, we compare the popularity of items by their *IPD* at a given point of time. We will describe, here, how to calculate the IPD of each video.

Fig. 1 illustrates the arrival of requests at the server for different objects with varying popularity. Each arrow represents a request arrival and the horizontal axis represents time. We define the j^{th} *backward distance in time (BDT)* of object i as the time since the j^{th} last arrival for object i, and we denoted this by $t_{i,j}$. For example in Fig. 1 the 3rd BDT of object 1, denoted by $t_{1,3}$, is the time from the 3rd to the last arrival of video 1. We define the inverted pyramid distance (IPD) of video i as the sum of last k BDTs of video i, and denote it by T_i. That is:

$$T_i = \sum_{i=1}^{k} t_{i,j}. \tag{4}$$

The request arrival rate for a popular video is high, and gaps between requests are low. When we calculate the IPD of a video, the time since the last request of the video is added k times, the time duration between the 2nd last request and the last request of that video is added $(k - 1)$ times, the time duration between the 3rd last request and the 2nd last request is added $(k - 2)$ times, and so on. Consequently, when IPDs are calculated, the importance of a request is de-emphasized in a *pyramid-like* fashion, depending on how old the request is. As we will show, the IPD of a video increases when the popularity of the video decreases.

If the video lacks popularity, or is newly introduced, and the total number of requests is not sufficiently large, Equation (4) cannot be used to calculate its T_i value. While it is the case that another formula with different weighting could be used to calculate a popularity estimate in this case. There are several reasons not to do so: 1. We could have less confidence in the estimate; 2. Choosing to shift an object to the cache should be a conservative decision as it consumes resources and so should be done with high confidence in the performance improvement; 3. Small sample size estimation might be subject to gaming by, for instance, video companies wishing to increase the perceived popularity of their products. These issues of conservative estimation, small sample size and the potential of gaming are related in a complex way and are not discussed further here. We will return to them in future work.

MIPD is applicable in scenarios where we need to identify the C most popular objects from a total of N objects and rank those C objects in order of their popularity. MIPD calculates the most recent k BDTs of each of the objects and uses it to calculate their IPDs. After that, MIPD sorts the objects in increasing order of their IPDs, corresponding to the decreasing order of their popularity. The value of k is selected based on the value of C, so that the estimated popularity rankings meet a pre-defined confidence level, at least for the C most popular objects.

The pseudo-code of the MIPD method is given in Algorithm 1. As described there, the inputs to the MIPD algorithm are k, C, the arrival time matrix $\mathbf{t}$, and the current time t_0. The arrival time matrix $\mathbf{t}$ is derived from the request arrival history as follows,

$$\mathbf{t}(i,j) = \begin{cases} 0, & \text{if there are no past requests for object } i \\ j^{\text{th}} \text{ last request time of video } i, & \text{otherwise.} \end{cases} \tag{5}$$

The vector $\mathbf{T}$ stores the IPDs of the videos. It is initialized with all of its elements set to 0. It can be seen from the pseudo-code that the complexity of the MIPD algorithm is linear in k. While Algorithm 1 shows a straightforward version of the MIPD algorithm, we can use a different implementation with lower overhead and memory requirements if the popularity estimation needs to be carried out repeatedly on the arrival of each new request, such as in cache replacement algorithms. We will discuss this implementation in Section III-C.

Algorithm 1 MIPD Algorithm

Input: $k, C, \mathbf{t}, t_0, m$
Output: C most popular objects sorted in order of their popularity

$\mathbf{T} \leftarrow 0_{N\times 1}$
$i \leftarrow 1$
$j \leftarrow 1$
while $i \leq N$ **do**
  if $m >= k$
  while $j \leq k$ **do**
    $\mathbf{T}(i) \leftarrow \mathbf{T}(i) + (t_0 - \mathbf{t}(i,j))$
    $j \leftarrow j + 1$
  end while
  $i \leftarrow i + 1$
  if $m < k$
  $\mathbf{T}(i) \leftarrow \sum_{j=1}^{m} t_{i,j} + (k-m)t_{i,m}$
end while
Sort $\mathbf{T}$ and index top C associated objects

B. CHOOSING THE VALUE K FOR MIPD

We show here that MIPD can accurately rank objects in order of popularity for an appropriate value of k, and discuss two methods that can be used to find suitable values for k: the first of these gives an exact minimal value for k bearing in mind the level of confidence we require, whereas the other allows us to calculate some rates of growths of upper bounds for k as the confidence level and size of cache increase. As discussed before, when IPDs are calculated, the importance of a request is de-emphasized in a pyramid-like fashion depending on how old the request is. This allows quick response to changes in video popularity. Even if the popularity of video objects changes frequently with time, such changes still occur in the order of hours, and such periods typically contain thousands of request arrivals. In the following analysis, we consider a small time window, during which the popularity of the objects can be assumed to remain stable. We assume, also, that the request arrival process is approximated by a pseudo-homogeneous Poisson process during this time. This is justifiable because VoD requests are normally generated by a large number of independent users [28], [29]; we will discuss this further in Section IV.

Let λ_i be the mean request arrival rate of video i. Note that the j^{th} BDT of object i is equivalent to a summation of j request inter-arrival times of object i. Since the request arrivals for video i follows a Poisson process with rate λ_i, the inter-arrival times follow an exponential distribution with mean $1/\lambda_i$. As a result, the j^{th} BDT of video i is Erlang distributed with scale parameter of λ_i and shape parameter j. In turn, the IPD is a summation of k such Erlang distributed BDTs, so that the IPD of video i is also Erlang with scale parameter λ_i and shape parameter K, where

$$K = \sum_{j=1}^{k} j = \frac{k(k+1)}{2}.$$

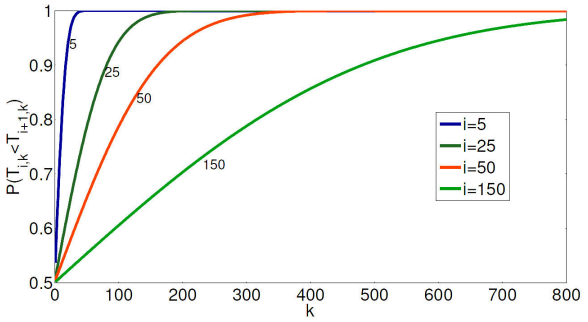


FIGURE 2. Variation of $P(T_{i,k} < T_{i+1,k})$ with k.

Recall that, in our proposed popularity estimation method, objects are ranked in decreasing order of popularity, by indexing them in increasing order of IPD. Consider two objects i and $i+1$, such that the popularity of object i is greater than the popularity of object $i+1$. Let $T_{i,k}$ and $T_{i+1,k}$ be the IPDs of the objects i and $i+1$, respectively. We can use these IPDs to correctly identify the more popular object out of these two only if $T_{i,k} < T_{i+1,k}$ with high probability. This probability is [30]:

$$P(T_{i,k} < T_{i+1,k}) = \int_0^{\infty} f_{T_{i+1,k}}(t) F_{T_{i,k}}(t) dt, \tag{6}$$

where $f(.)$ and $F(.)$ are the PDF and Cumulative Distribution Function (CDF) of the Erlang distributed random variables $T_{i,k}$ and $T_{i+1,k}$, respectively.

Theorem 1: The probability of IPD of object i being less than the IPD of object $i+1$ is given by:

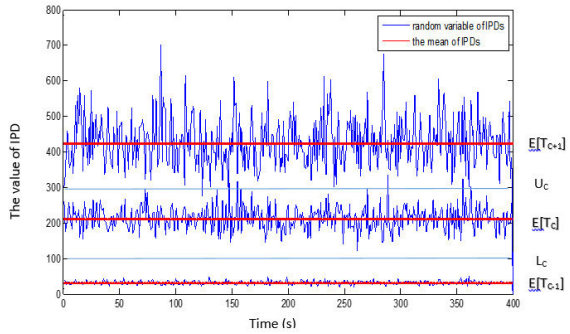
$$P(T_{i,k} < T_{i+1,k}) = I_{p_i}(K, K), \tag{7}$$

where $p_i = \frac{\lambda_i}{\lambda_i + \lambda_{i+1}}$, $K = \frac{k(k+1)}{2}$, and $I_{p_i}(.)$ is the regularized Beta function.

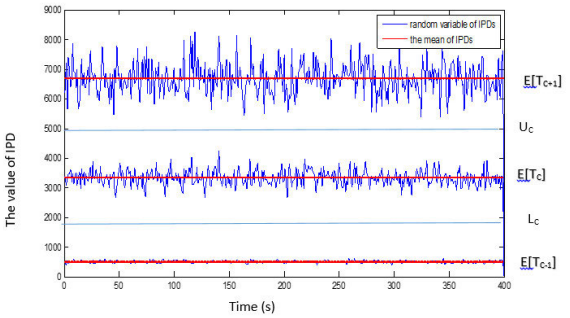
Proof: See Appendix A. □

Now, let i and $i+1$ be two videos with consecutive popularity ranks, where the popularity of i is greater than the popularity of $i+1$. For this scenario, we plot the variation of $P(T_{i,k} < T_{i+1,k})$ as a function of k in Fig. 2, for a range of values of i. As can be seen from Fig. 2, the larger k is, the closer the value of $P(T_{i,k} < T_{i+1,k})$ is to 1. This indicates that for k sufficiently large, the more popular object out of i and $i+1$ can be identified with confidence based on their IPDs. Moreover, as can be seen from Fig. 2, for a given k, $P(T_{i,k} < T_{i+1,k})$ is closer to unity for popular objects (e.g., $i = 5$) than for less popular objects (e.g., $i = 150$). In other words, for a given k, IPD-based popularity estimation gives better performance for popular objects.

Intuitively, the value of k; that is, the number of request arrival times from each of the videos used for the popularity estimation, should depend on the number of objects that we need to rank with a given accuracy. Suppose we need to identify and rank the C most popular objects from a total of N. As discussed in the preceding subsection, for a given k,



(a) k=50



(b) k=200

FIGURE 3. Graphical representation of IPD noise.

IPD-based popularity estimation is more robust for popular objects than for less popular objects. More specifically, for a given k, $P(T_{C,k} < T_{C+1,k}) \geq 1-\delta$ implies that $P(T_{i,k} < T_{i+1,k}) \geq 1-\delta$ for all $1 \leq i \leq C$. As a result, a value of k large enough to identify the popular object out of the two objects with popularity ranks C and $C+1$ with a certain level of confidence, is large enough to rank all the other objects with popularity ranks less than C, with the same if not higher confidence.

Our exact method to find a suitable value for k is based on (24). We define a confidence bound δ and find the value of k that satisfies:

$$I_{p_c}\left(\frac{k(k+1)}{2}, \frac{k(k+1)}{2}\right) \geq 1-\delta. \tag{8}$$

The formula in (8), while relatively easy to compute using, say, Matlab, gives little or no insight, into the analytical form of the value of k as a function of the confidence level $(1-\delta)$ and the size C of the cache, or at least of λ_C. Such an expression is available provided we acccept an approximate method for computing k. This, at least in principle, gives such an analytical expression. Our approach is as follows. The IPDs are random variables, each taking different values with varying probabilities, and scattered around their means. This statistical noise of IPDs can affect popularity estimation and result in errors in the ranking. Popularity estimation becomes more robust if gaps between IPDs are increased. Our proposed approximate method to determine the value of k is based on this observation. A lower bound L_C, and an upper bound U_C for the IPD of C^{th} popular object are defined in Eq. (9), and we select k for the algorithm such that this IPD satisfies these bounds with high probability. By doing so, we ensure that the gaps between the IPD of the C^{th} most popular object and the two nearest IPDs, i.e., the IPDs of the $(C+1)^{\text{th}}$ and the $(C-1)^{\text{th}}$ popular objects, are sufficiently large. This is graphically illustrated in Fig. 3(a) and Fig. 3(b).

$$L_C = \frac{E[T_{C,k}]+E[T_{C-1,k}]}{2}, \text{ and } U_C = \frac{E[T_{C,k}]+E[T_{C+1,k}]}{2}. \tag{9}$$

Recall that the IPD of each object is Erlang distributed. In the first subfigure of Fig. 3(a), we have selected $k = 50$. The gap between the IPD of the C^{th} most popular object and $(C-1)^{\text{th}}$ popular object is sufficiently large, while the gap between the IPD of the C^{th} most popular object and the $(C+1)^{\text{th}}$ most popular object is not sufficiently large. In the second subfigure, we have selected $k = 200$. The gaps between the IPD of the C^{th} most popular object and the two nearest IPDs are sufficiently large in this case. These two graphs demonstrate that for larger k, better performance of the popularity estimation is achieved.

Definition 1: The IPD-based popularity estimation method ranks C most popular objects in order of their popularity with a confidence of $1-\delta$ *if,*

$$\max\left(P(T_{C,k} \leq L_C), P(T_{C,k} \geq U_C)\right) \leq \delta. \tag{10}$$

Note that $1-\delta$ measures the confidence of popularity estimation with respect to the least popular video of interest. For example, if we need to identify the 100 most popular videos to store in a cache server, $1-\delta$ measures how good the popularity estimation of the 100^{th} most popular video. The confidence level of the popularity ranking of each video more popular than this one will be higher when its popularity is estimated using IPDs. We see then that $1-\delta$ provides a lower bound for the confidence of the overall popularity estimation. It is enough, then, to consider popularity of higher ranked videos to achieve good performance, though overall performance can be further improved by decreasing δ.

As the IPD $T_{C,k}$ of the C^{th} most popular object is Erlang distributed with scale λ_C and shape $K = k(k+1)/2$, its expected value is:

$$E[T_{C,k}] = \frac{K}{\lambda_C}. \tag{11}$$

We use Chernoff bounds for $T_{C,k}$ to further simplify Eq. (10). As proved in Appendix B, the Chernoff Bounds for the random variable $T_{C,k}$ are given by:

$$P(T_{C,k} \leq L_C) \leq e^{-L_C\lambda_C}\left(\frac{eL_C\lambda_C}{K}\right)^K, \tag{12}$$

$$P(T_{C,k} \geq U_C) \leq e^{-U_C\lambda_C}\left(\frac{eU_C\lambda_C}{K}\right)^K. \tag{13}$$

Now, substituting (12) and (13) in (10), we see that the values of k that satisfies:

$$\max\left[e^{-L_C\lambda_C}\left(\frac{eL_C\lambda_C}{K}\right)^K, e^{-U_C\lambda_C}\left(\frac{eU_C\lambda_C}{K}\right)^K\right] \leq \delta, \tag{14}$$

also satisfies Eq. (10). Solving Eq. (14) for K gives:

$$K \geq \max\left(\frac{\ln(\delta)}{\frac{1}{2}-\frac{\lambda_C}{2\lambda_{C-1}}+\ln\left(\frac{1}{2}+\frac{\lambda_C}{2\lambda_{C-1}}\right)}, \frac{\ln(\delta)}{\frac{1}{2}-\frac{\lambda_C}{2\lambda_{C+1}}+\ln\left(\frac{1}{2}+\frac{\lambda_C}{2\lambda_{C+1}}\right)}\right). \quad (15)$$

We note that, for the Zipf distribution, and $C > 1$,

$$1-\frac{\lambda_C}{\lambda_{C-1}} > \frac{\lambda_C}{\lambda_{C+1}} - 1 > 0,$$

and this combined with a simple derivatives argument for the function $\phi(x) = \frac{1}{2}(1-x) + \ln\left(\frac{1+x}{2}\right)$ shows that the second expression in the max in (15) is always larger so that the correct choice of k is the least for which

$$\frac{k(k+1)}{2} = K \geq \frac{\ln(\delta)}{\frac{1}{2}-\frac{\lambda_C}{2\lambda_{C+1}}+\ln\left(\frac{1}{2}+\frac{\lambda_C}{2\lambda_{C+1}}\right)}. \quad (16)$$

We observe that k found in this way increases as the square root of $\ln(\delta)$. Numerical simulations suggest that, as a function of C, the value of k, calculated by this method, increases by less than $C\log(C)$ and more than $C\sqrt{\log(C)}$, so slightly greater than linearly.

C. OPTIMISATION OF MIPD FOR CACHE REPLACEMENT

In Section III-A, we presented a straightforward implementation of MIPD in Algorithm 1. This implementation is suitable for scenarios where popularity does not need frequent re-estimation. However, some cache servers are updated whenever a user requests a video not already stored in the cache. When these cache updates take place, the least popular objects are identified and removed from the cache if the remaining cache storage space is insufficient to accommodate the newly requested object. Hence, content management algorithms for cache servers need to estimate popularity at every new request arrival. For rapid cache replacement, the IPD algorithm needs to be implemented with low computational overhead.

We now discuss how to reduce the overhead and memory requirements of MIPD to carry out popularity estimation on the arrival of each new request. Let $T_{i,k}^{(\tau)}$ denote the IPD of object i at τ seconds. Here, τ is the time at which the previous request arrived at the cache server, which coincides with the time of the previous popularity estimation. Suppose that a new request for object h arrives at the cache server at $\tau+\Delta\tau$ seconds. We need to re-estimate the popularity of the objects at $\tau+\Delta\tau$ seconds by recalculating the IPD of each of the objects. Note that when the popularity of the objects are estimated at $\tau+\Delta\tau$ seconds, the IPD update for each of the objects is:

$$T_{i,k}^{(\tau+\Delta\tau)} = \begin{cases} (\tau+\Delta\tau-\tau_1^{(i)})+(\tau+\Delta\tau-\tau_2^{(i)}) & \\ \quad +\cdots+(\tau+\Delta\tau-\tau_{k-1}^{(i)}) & \text{if } i=h \\ (\tau+\Delta\tau-\tau_1^{(i)})+(\tau+\Delta\tau-\tau_2^{(i)}) & \\ \quad +\cdots+(\tau+\Delta\tau-\tau_k^{(i)}) & \text{otherwise,} \end{cases} \quad (17)$$

where $\tau_j^{(i)}$ is the j^{th} last request arrival time of object i with respect to time τ. In Eq. (17), only the last $(k-1)$ request arrival times are considered when calculating the IPD of object h, since the most recent request for the object h arrived at $\tau+\Delta\tau$ seconds. On the other hand, the last k arrival times of each of the objects are considered when IPDs are Eq. calculated for the other objects. Rewriting (17), we see that:

$$T_{i,k}^{(\tau+\Delta\tau)} = \begin{cases} T_{i,k}^{(\tau)}+k\Delta\tau-(\tau+\Delta\tau-\tau_k^{(i)}), & \text{if } i=h \\ T_{i,k}^{(\tau)}+k\Delta\tau, & \text{otherwise.} \end{cases} \quad (18)$$

By (18), we need only the current IPDs and the k^{th} last arrival time of each object, to re-calculate the new IPDs on receiving a new request. The k^{th} last arrival time of each object can be maintained in the form of *first in first out (FIFO)* queues, each with a length of k. When a request arrives, the relevant FIFO queue is updated with the current time, while removing the last entry in the queue. The ejected entry is the $\tau_k^{(i)}$ in Eq. (18). It should be clear then that IPD-based object popularity estimation can be carried out using Eq. (18) with low overhead, making it a suitable cache replacement algorithm.

IV. VOD SYSTEM MODELING

A. POPULARITY DISTRIBUTION IN VOD SYSTEM

We now consider a wide range of real-life issues. We discuss many recently references and data-sets, and in particular, we design our experiments based on what we learn from studies that use real-life time-dependent video arrival processes. We design our experiments to follow realistic data traffic models. According to [11], [31], the distribution of the video popularity follows *Pareto's Law*. Namely, this distribution is *heavy − tailed* which implies that very few popular movies dominate most of the population of the requested videos. This assertion has also been supported by Choi *et al.* [32], that models the prevalence of videos by the *Zipf* distribution (which is also *heavy-tailed*). They propose that, among N videos, the popularity of video i, where i is the rank of popularity in this VoD system, has request frequency of W/i (W is a normalization constant satisfying the equation). And all the videos should satisfy:

$$\sum_{i=1}^{N} \frac{W}{i^{\alpha}} = 1 \quad (19)$$

where α is the parameter of the distribution.

Fig. 4 shows the relationship between the frequency of request and the popularity rank of videos, which is modeled by the *Zipf* distribution.

B. REQUEST ARRIVAL PROCESS IN VOD SYSTEM

Many research results on experiments using real VoD systems (e.g., [28], [29], [33]) show that video request traffic tends to follow a Poisson process over a sufficiently short period. Liu *et al.* [28] observed VoD user behaviors over a period of five months, and their research results provide a reliable basis

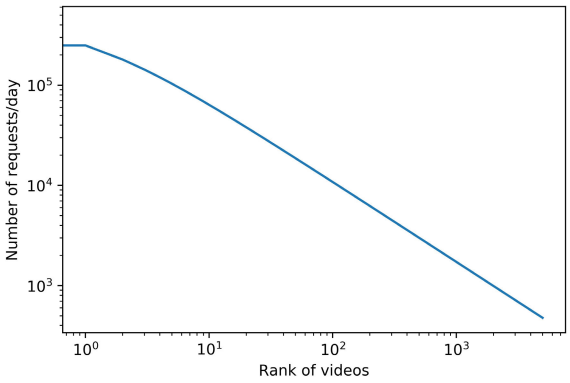


FIGURE 4. *Zipf* distribution ($\alpha = 0.8$) for video popularity.

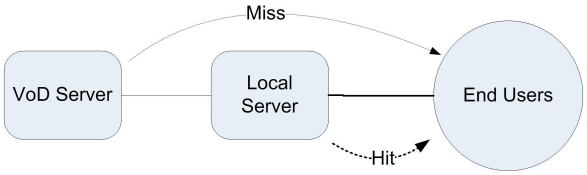


FIGURE 5. VoD system architecture considered for the evaluation.

TABLE 1. The parameter of VoD model.

Parameter	Value
Number of videos in VoD system (N)	5000
Number of most popular videos (C)	200
Parameter of the Zipf distribution (α)	0.8
Total requests to the VoD system per day (λ)	10,000,000 req/day

for simulating VoD systems. They indicate that the request traffic for a VoD system can be approximated by a pseudo-stationary Poisson process over a time interval of about thirty minutes. Tanzil *et al.* [33], in order to generate user request data based on real-world YouTube reports, also assume that the request arrival process to a VoD server follows a Poisson process during a specified time slot (around twenty minutes). A pseudo-stationary Poisson process is defined by

$$P(X = k) = \frac{\lambda^k e^{-\lambda t}}{k!}, \quad k = 0, 1, 2, \ldots \tag{20}$$

where k is the request count in a time period of length t, and λ is the arrival rate which, in our case, changes every half an hour.

The probability distribution of the time between two requests is exponentially distributed with probability density function

$$f(x; \lambda) = \begin{cases} \lambda e^{-\lambda x}, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{21}$$

where x is the time elapsed between two requests. For this (continuous) random variable, λ is the mean arrival rate and $1/\lambda$ is the mean time between two requests. Accordingly, we will use a time-dependent Poisson process that relies on the studies in [29], [31], [34] for parameter fitting.

C. PARAMETERS SETTING IN SIMULATION

A single level VoD system with one VoD server and a *local server* is considered for our evaluation. Fig. 5 illustrates graphically the logical architecture of this simulated network. In our simulations, we estimate the popularity of the videos at the VoD server and pre-place the *full* videos at the local server based on their estimated popularity. As shown in Fig. 5, when a user requests a video, the local server can send out the video to the user if the requested video is pre-placed in it. Otherwise, the request is served by the VoD server. Moreover, we assume that the video sessions run the full length of the videos. This is, however, not the case in real VoD systems as shown by recent research [11], [17], [18]. Nevertheless, this assumption does not affect the validity of our evaluation, as we are only concerned with the popularity estimation aspect of content placement. Of course, we can achieve better performance by pre-placing partial videos, but this is not the focus of our work.

In our VoD system model, the total number of videos is $N = 5000$. According to the Zipf distribution, around 20% of the 5000 videos occupy 80% of the requests in a day while the rest of the videos have far fewer requests. The cache size was selected to be 200 videos. Since the video request process is pseudo-Poisson, we divide a day into 48 half-hour periods, while the simulation continues for 24 hours. According to [28], the arrival rate of the requests varies over a day, and the daily pattern of the request traffic remains similar across different days. Also diurnal request patterns vary across movies. More specifically, some videos are more popular in the morning (e.g., cartoons and morning news), some videos are more popular during the noon break time (e.g., news) and other videos are more popular in the evening (e.g., movies, drama). This behavior of the request traffic in VoD systems is also reported in [35], [36]. For this reason, the request counts for popularity ranked movies in an hour may not follow a Zipf distribution. So we can assume the request counts for popularity ranked movies in a whole day follow Zipf distribution.

We classify these 5000 videos into three classes (Class_M, Class_A, Class_E) according to which part of the day (morning, afternoon, evening, respectively) their popularity is highest.

Let λ_i be the request arrival rate for video i averaged over a day, $i = 1, 2, \ldots, N$ and $\lambda_{i,j}$ the request arrival rate for movie i in the j^{th} half hour, $j = 1, 2, \ldots, 48$. Then we have $\lambda_i = \sum_{j=1}^{48} \lambda_{i,j}$, where each λ_is follows a Zipf distribution. According to [28], the average number of hourly requests is around 10^6 requests per VoD system of around 250,000 videos, so in this case $\sum_{i=1}^{N} \lambda_i = 10{,}000{,}000$.

The parameters of the VoD system and their meanings and defaults are provided in Table 1.

V. PERFORMANCE EVALUATION

We describe here the simulation experiments to evaluate the performance of our proposed MIPD popularity estimation algorithm. To understand how k influences the accuracy and efficiency of MIPD, we run the experiments with different values of k. We also compare the performance of MIPD against other common popularity estimation methods: LFU, LRU, LRFU and EWMA. It is assumed that the popularity of the videos is re-evaluated, and the pre-placed content is updated accordingly, every half hour. By default, we use $k = 456$, the value given by the first method, for comparison. This value is determined via the approximate method described in Section III-B, so as to rank the 5000 objects with confidence greater than $1-\delta = 0.9$. For the LRFU popularity estimation method, using $\tau = 0.001$ in Eq. (1) has given the best performance for our situation and so this value of the smoothing parameter is used for LRFU in our simulations. For the LFU method, we assume that the entire history is available for popularity estimation, namely, 24 hours in the current scenario. Moreover, we assume that the recency of the most recent request of each video is used for the LRU method.

A. THE EFFECT OF K ON MIPD PERFORMANCE

We use the Hit ratio (HR) and the Rank Bias (RB) as the performance metrics in our evaluation. We consider a scenario where video popularity ranks estimated at the VoD server are used to determine the videos to be pre-placed at the local server, to maximized HR. An efficient popularity estimation method should accurately identify the latest trends on user demands, and therefore should yield better HR at the local server. hit ratio (HR) is defined as:

$$\text{HR} = \frac{\text{Number of requests served by the local server}}{\text{Total number of videos requested}}. \quad (22)$$

For a *large* sample of N videos requests, the number of requests for the i^{th} most popular video is approximately $p_i N$, where p_i is the request probability of the i^{th} most popular video. The HR expected at the local server is:

$$\text{HR} \approx \sum_{i \in \mathbf{S}} p_i, \quad (23)$$

where $\mathbf{S}$ is the set of popularity ranks of the videos that are pre-placed. Given a local server with constrained storage capacity, we should then pre-place the most popular videos in order to achieve the maximum HR.

As shown in Fig. 6, HRs of different k fluctuate through a day because the popularity of each of the various classes of videos changes according to different patterns. Additionally, as k increases, the speed of response of MIPD to changing popularity decreases, because larger k means MIPD will take account of more of the historical requests. More specifically, as shown in Fig. 7, with increasing k, the accuracy of MIPD first increases gradually and then shows a downward trend: larger k does not guarantee better performance, especially when the popularity rank of videos changes frequently.

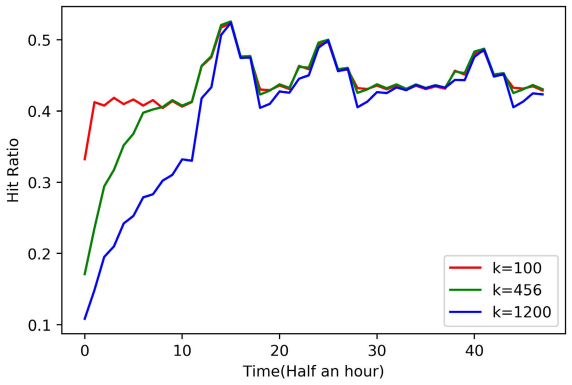


FIGURE 6. Variation of Hit Ratio with time for different value of k.

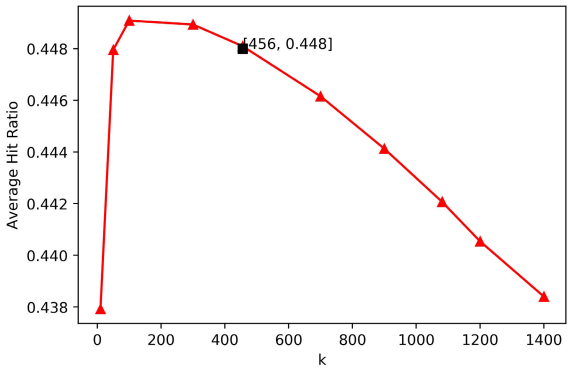


FIGURE 7. Average hit ratio with different k.

We also use RB as performance metrics in this paper; this is the absolute deviation between the actual rank and the estimated rank of a specific video. In this simulation, we assume that the diurnal request pattern of each of the popularity types remains constant, so that their ranking will not change during the day. In order to compare the different value of k through RB, we introduce three videos into the system with known popularity rank of 2, 20 and 200 at the 20^{th} half an hour. As mentioned before, popularity of videos is estimated every half hour.

Fig. 8 shows that when a new video (for which the popularity rank is around 200) is introduced, it ranks very low in popularity as it has no record of requests. When the new video are subsequently requested by users, accuracy of popularity estimation increases gradually. The black dashed lines in the figures are the actual popularity rank of the video. For the different values of k, the estimated popularity fluctuates around the true popularity over time. Variation of the value of k changes the responsiveness and accuracy of MIPD. As k increases, the bias of MIPD decreases gradually suggesting that the accuracy of MIPD is improved. On the other hand, the speed of response of MIPD shows a downward trend.

To investigate the accuracy of MIPD for different popularity ranked videos, we accumulate the absolute deviation through time after new videos were introduced into system,

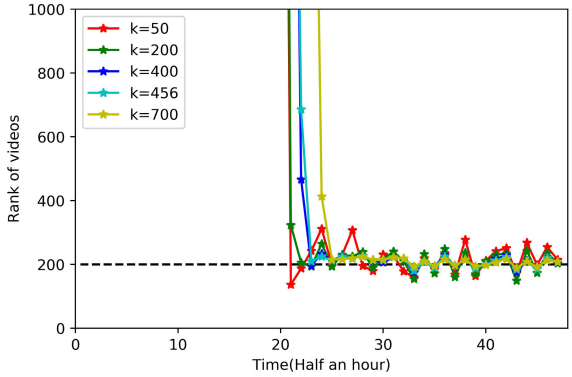


FIGURE 8. Variation of Rank200 bias with time.

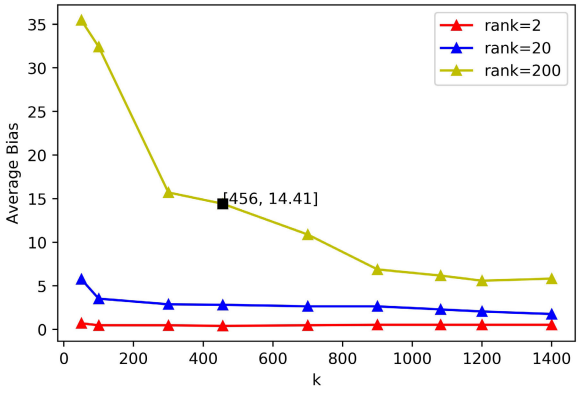


FIGURE 9. Average Bias with different k.

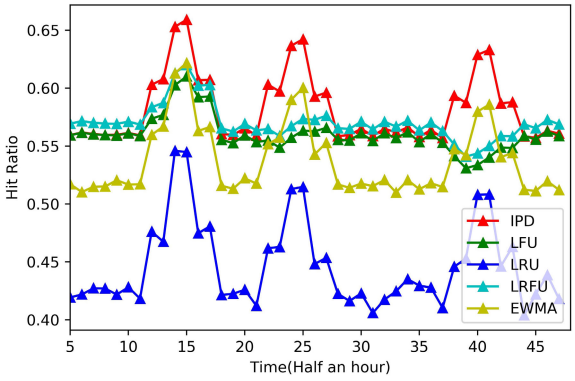


FIGURE 10. Variation of Hit Ratio with time for different methods.

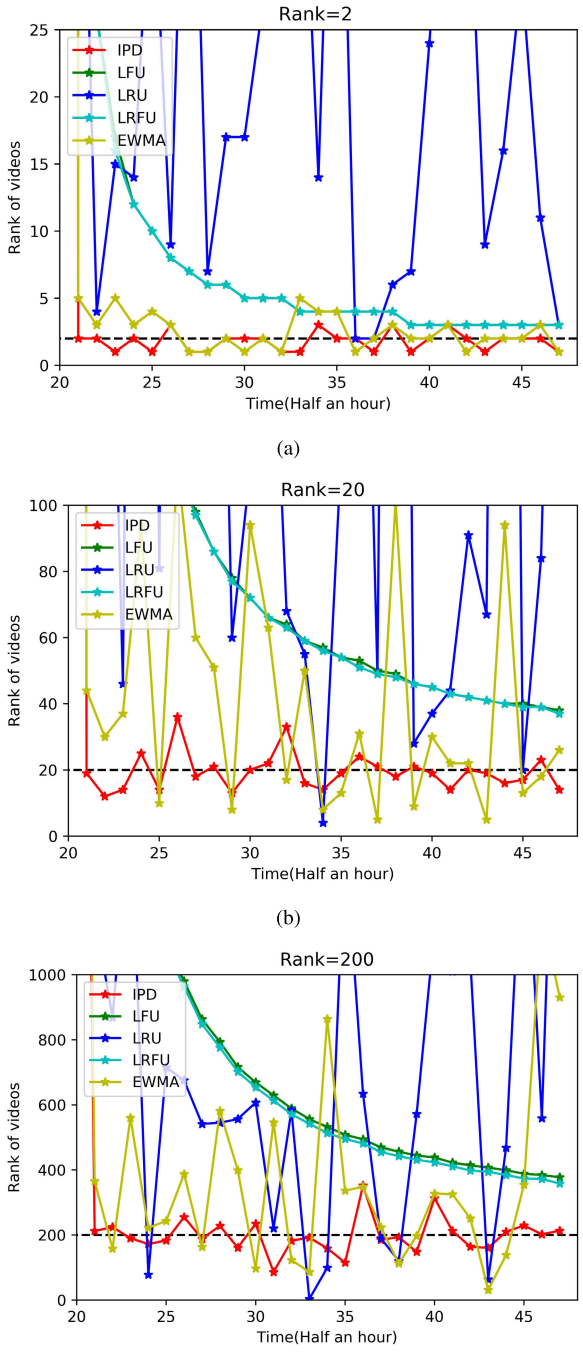


FIGURE 11. Variation of Rank bias with time for different methods.

and compare the variation in ranking of videos. As Fig. 9 shows, the higher the popularity of a video, the more accurate the MIPD rank estimate. The average absolute deviation of a video with popularity $rank = 2$ is smaller than that for a video with popularity $rank = 20$. For a video with popularity $rank = 200$, the average absolute deviation is much larger.

B. COMPARISON BETWEEN DIFFERENT METHODS

In this subsection, we also use HR and RB as the performance metrics in our evaluation. We use $k = 456$, noting that speed of response to variation of popularity rank with time is not a major issue, because the 200 most popular videos to be put in the cache have a high enough request rate. We compare the HRs achieved by the local server under each of the considered popularity estimation method, when there are three different classes of videos with varying popularity patterns over the day. We compare the accuracy for ranking the newly introduced videos of these five methods.

Fig. 10 shows that the HRs of the five methods fluctuate throughout the day. Note that we record the HR from $t = 5$

(half hour) because, before that time, there are insufficient request records to calculate the HR. It is clear that MIPD achieves the best performance. MIPD has a higher HR than the other four methods most of the time, especially when the popularity of videos is changing. This suggests that MIPD can respond very effectively to popularity changes of videos. LRU has the lowest HR, despite a rapid response to popularity change. The HRs of LRFU and LFU methods are also considerable, even becoming higher than MIPD at some times, but their response to popularity change is relatively slow. The reason is that LRFU and LFU methods take more account of historical requests than other methods. The EWMA method also shows a good performance in simulation: it has a higher HR than LRU, and responds faster than LFU and LRFU when the popularity of videos change.

We also use RB to compare the different popularity methods, as shown in Figs. 11(a)–11(c). When new videos are introduced into the library at $t = 20$ (half an hour), none of the methods can identify the popularity rank of those new videos promptly because of insufficient data.

Figs. 11(a)–11(c) show the variations of the estimated video popularity ranking with time for three videos with different popularities. The statistical variability in the popularity ranking using MIPD appears less than in other methods, regardless of whether the rank is $k = 2$, $k = 20$ or $k = 200$. The performance of LRU is the worst, consistent with its poor HR. The performances of both LFU and LRFU are good, and as request statistics become sufficient, their accuracy becomes higher. It also indicates that LFU and LRFU can perform much better when the popularity of videos is stable. EWMA also makes a relatively accurate prediction of the popularity rank of videos and its fluctuations, though larger than those of MIPD, are relatively small.

VI. CONCLUSION

In this paper, we have described a significant extension (called MIPD) of the IPD method for object popularity estimation reported in [14], where it was called the Last-k method. MIPD, derived from the arrival times of requests for each of the objects, de-emphasizes the importance of a request in a pyramid-like fashion, according to how old the request is. The popularity of a video reflected by its IPD is a seamless blend of long-term and short-term popularity of that video. We analyzed the relationship between the value of the parameter k in MIPD and confidence in the estimation, and demonstrated how to find a large enough k to achieve a given confidence requirement. We also discuss how to reduce the overhead and memory requirements of the algorithm. The implementation can refresh the popularity rank efficiently making it suitable in scenarios where the popularity of the objects are required to be estimated very frequently. We use a range of simulations to justify MIPD in comparison with four other popularity estimation methods proposed in the literature. The simulation results reveal that MIPD popularity estimation method outperforms the other four methods according to both hit rate (HR) and rank bias (RB).

APPENDIXES

APPENDIX A
PROOF OF THEOREM I

We restate *Theorem 1* here: The probability of IPD of object i being less than the IPD of object $i+1$ is given by:

$$P(T_{i,k} < T_{i+1,k}) = I_{p_i}(K, K),$$

where $p_i = \frac{\lambda_i}{\lambda_i+\lambda_{i+1}}$, $K = \frac{k(k+1)}{2}$, and $I_{p_i}(.)$ is the regularized Beta function.

Substituting the relevant expressions in the equation above and simplifying, we have:

$$\begin{aligned} P(T_{i,k} < T_{i+1,k}) &= \int_0^\infty f_{T_{i+1,k}}(t) F_{T_{i,k}}(t) dt \\ &= \int_0^\infty \frac{\lambda_{i+1}^K t^{K-1} e^{-\lambda_{i+1}t}}{(K-1)!} \left(1 - \sum_{j=0}^{K-1} e^{-\lambda_i t} \frac{(\lambda_i t)^j}{j!}\right) dt. \end{aligned} \quad (24)$$

Using the integral identities from [37], yields a further simplification of Eq. (24):

$$P(T_{i,k} < T_{i+1,k}) = 1 - (1-p_i)^K \sum_{j=0}^{K-1} p_i^j \binom{j+K-1}{k}, \quad (25)$$

where $p_i = \frac{\lambda_i}{\lambda_i+\lambda_{i+1}}$. Let X be a negative binomial random variable with parameters K and p_i, i.e., $X \sim NB(K, p_i)$, and $F_{K,p_i}(x)$ denote the cumulative distribution function (CDF) of X. Then, Eq. (25) can be reduced to:

$$\begin{aligned} rClP(T_{i,k} < T_{i+1,k}) &= 1 - \sum_{j=0}^{K-1} P(X = j) \\ &= 1 - F_{K,p_i}(K-1). \end{aligned} \quad (26)$$

Using the expression for the CDF of the negative binomial distribution [38], we can write:

$$F_{K,p}(K-1) = 1 - I_{p_i}(K, K), \quad (27)$$

where $I_{p_i}(.)$ is the regularized Beta function. Finally, by substituting Eq. (27) in (26), we have

$$P(T_{i,k} < T_{i+1,k}) = I_{p_i}(K, K), \quad (28)$$

which completes the proof of Theorem 1.

APPENDIX B
CHERNOFF BOUNDS FOR ERLANG RANDOM VARIABLES

The Chernoff Bounds for a random variable X are given by:

$$P(X \le L) \le \inf_{t<0} e^{-tL} M_X(t), \quad (29)$$

and

$$P(X \ge U) \le \inf_{t>0} e^{-tU} M_X(t), \quad (30)$$

where $M(t) = E[e^{tX}]$ is the moment generating function of X. The moment generating function of an Erlang distributed random variable with scale parameter λ and shape parameter K, is [39]:

$$M(t) = \left(\frac{\lambda}{\lambda - t}\right)^K. \tag{31}$$

Substituting (31) in (29) and (30), gives the Chernoff bound for the Erlang distributed random variable $T_{C,K}$ as:

$$P(T_{C,K} \leq L) \leq \inf_{t<0} e^{-tL} \left(\frac{\lambda}{\lambda - t}\right)^K, \tag{32}$$

and

$$P(T_{C,K} \geq U) \leq \inf_{t>0} e^{-tU} \left(\frac{\lambda}{\lambda - t}\right)^K. \tag{33}$$

By differentiating (32) and (33) with respect to t and setting the result to 0, we see that the values of t that minimize the right hand sides of (32) and (33) are,

$$t_L^* = \lambda - \frac{K}{L}, \tag{34}$$

and

$$t_U^* = \lambda - \frac{K}{U}, \tag{35}$$

respectively. Finally, substituting Eq. (34) and Eq. (35), in (32) and (33), respectively, we obtain:

$$P(T_{C,k} \leq L) \leq e^{-L\lambda} \left(\frac{eL\lambda}{K}\right)^K, \tag{36}$$

and

$$P(T_{C,k} \geq U) \leq e^{-U\lambda} \left(\frac{eU\lambda}{K}\right)^K. \tag{37}$$

REFERENCES

[1] (Feb. 2019). *Cisco Visual Networking Index: Forecast and Trends, 2017–2022 White Paper*. Accessed: Jun. 10, 2019. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html

[2] M.-C. Huang, C.-H. Chang, C.-W. Tseng, and R.-J. Lee, "Content delivery based on popularity and time slot," in *Proc. 18th Int. Conf. Parallel Distrib. Comput., Appl. Technol. (PDCAT)*, Dec. 2017, pp. 344–347.

[3] Y. Zhang, C. Gao, Y. Guo, K. Bian, X. Jin, Z. Yang, L. Song, J. Cheng, H. Tuo, and X. Li, "Proactive video push for optimizing bandwidth consumption in hybrid CDN-P2P VoD systems," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2018.

[4] D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. K. Ramakrishnan, "Optimal content placement for a large-scale VoD system," *IEEE/ACM Trans. Netw.*, vol. 24, no. 4, pp. 2114–2127, Aug. 2016.

[5] S.-B. Lee, G.-M. Muntean, and A. Smeaton, "Performance-aware replication of distributed pre-recorded IPTV content," *IEEE Trans. Broadcast.*, vol. 55, no. 2, pp. 516–526, Jun. 2009.

[6] F. Thouin and M. Coates, "Video-on-demand networks: Design approaches and future challenges," *IEEE Netw.*, vol. 21, no. 2, pp. 42–48, Mar. 2007.

[7] G. Zhang, W. Liu, X. Hei, and W. Cheng, "Unreeling Xunlei Kankan: Understanding hybrid CDN-P2P video-on-demand streaming," *IEEE Trans. Multimedia*, vol. 17, no. 2, pp. 229–242, Feb. 2015.

[8] M. R. Garey, D. S. Johnson, and R. Sethi, "The complexity of flowshop and jobshop scheduling," *Math. Oper. Res.*, vol. 1, no. 2, pp. 117–129, May 1976.

[9] T. Qiu, Z. Ge, S. Lee, J. Wang, J. Xu, and Q. Zhao, "Modeling user activities in a large IPTV system," in *Proc. 9th ACM SIGCOMM Conf. Internet Meas. Conf. (IMC)*, 2009, pp. 430–441.

[10] D. De Vleeschauwer and K. Laevens, "Performance of caching algorithms for IPTV on-demand services," *IEEE Trans. Broadcast.*, vol. 55, no. 2, pp. 491–501, Jun. 2009.

[11] H. Yu, D. Zheng, B. Y. Zhao, and W. Zheng, "Understanding user behavior in large-scale video-on-demand systems," *ACM SIGOPS Oper. Syst. Rev.*, vol. 40, no. 4, p. 333, Oct. 2006.

[12] T. Trzcinski and P. Rokita, "Predicting popularity of online videos using support vector regression," *IEEE Trans. Multimedia*, vol. 19, no. 11, pp. 2561–2570, Nov. 2017.

[13] Y. Zhou, L. Chen, C. Yang, and D. M. Chiu, "Video popularity dynamics and its implication for replication," *IEEE Trans. Multimedia*, vol. 17, no. 8, pp. 1273–1285, Aug. 2015.

[14] C. Jayasundara, A. Nirmalathas, E. Wong, and N. Nadarajah, "Popularity-aware caching algorithm for video-on-demand delivery over broadband access networks," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2010, pp. 1–5.

[15] A. Sarhan, A. M. Elmogy, and S. M. Ali, "New Web cache replacement approaches based on internal requests factor," in *Proc. 9th Int. Conf. Comput. Eng. Syst. (ICCES)*, Dec. 2014, pp. 383–389.

[16] S. Podlipnig and L. Böszörmenyi, "A survey of Web cache replacement strategies," *ACM Comput. Surv.*, vol. 35, no. 4, pp. 374–398, Dec. 2003.

[17] B. Yin, J. Cao, Y. Kang, X. Lu, and X. Jiang, "A novel POMDP-based server RAM caching algorithm for VoD systems," *Multimedia Tools Appl.*, vol. 77, no. 10, pp. 13023–13045, May 2018.

[18] K.-W. Hwang, D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, V. Misra, K. K. Ramakrishnan, and D. F. Swayne, "Leveraging video viewing patterns for optimal content placement," in *Proc. Int. Conf. Res. Netw.* Berlin, Germany: Springer, 2012, pp. 44–58.

[19] G. Paschos, E. Bastug, I. Land, G. Caire, and M. Debbah, "Wireless caching: Technical misconceptions and business barriers," *IEEE Commun. Mag.*, vol. 54, no. 8, pp. 16–22, Aug. 2016.

[20] D. Swain, S. Marar, N. Motwani, V. Hiwarkar, and N. D. Valakunde, "CWRP: An efficient and classical weight ranking policy for enhancing cache performance," in *Proc. 4th Int. Conf. Image Inf. Process. (ICIIP)*, Dec. 2017, pp. 1–6.

[21] M. S. A. Khaleel, S. E. F. Osman, and H. A. N. Sirour, "Proposed ALFUR using intelegent agent comparing with LFU, LRU, SIZE and PCCIA cache replacement techniques," in *Proc. Int. Conf. Commun., Control, Comput. Electron. Eng. (ICCCCEE)*, Jan. 2017, pp. 1–6.

[22] S. Bai, X. Bai, and X. Che, "Window-LRFU: A cache replacement policy subsumes the LRU and window-LFU policies," *Concurrency Comput., Pract. Exper.*, vol. 28, no. 9, pp. 2670–2684, Jun. 2016.

[23] Y. Wang, Y. Yang, C. Han, L. Ye, Y. Ke, and Q. Wang, "LR-LRU: A PACS-oriented intelligent cache replacement policy," *IEEE Access*, vol. 7, pp. 58073–58084, 2019.

[24] S. J. Taher, "Performance evaluation of caching techniques for video on demand workload in named data network," Ph.D. dissertation, Univ. Utara Malaysia, Changlun, Malaysia, 2016.

[25] D. Lee, J. Choi, J.-H. Kim, S. Noh, S. L. Min, Y. Cho, and C. S. Kim, "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.

[26] S. Jin and A. Bestavros, "Popularity-aware greedy dual-size Web proxy caching algorithms," in *Proc. 20th IEEE Int. Conf. Distrib. Comput. Syst.*, Nov. 2002, pp. 254–261.

[27] Q. Ling, L. Xu, J. Yan, and Y. Zhang, "An adaptive caching algorithm suitable for time-varying user accesses in VOD systems," *Multimedia Tools Appl.*, vol. 74, no. 24, pp. 11117–11137, Dec. 2015.

[28] N. Liu, H. Cui, S.-H.-G. Chan, Z. Chen, and Y. Zhuang, "Dissecting user behaviors for a simultaneous live and VoD IPTV system," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 10, no. 3, pp. 1–16, Apr. 2014.

[29] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "YouTube traffic characterization: a view from the edge," in *Proc. 7th ACM SIGCOMM Conf. Internet Meas. (IMC)*, 2007, pp. 15–28.

[30] M. Zukerman, "Introduction to queueing theory and stochastic teletraffic models," 2013, *arXiv:1307.2968*. [Online]. Available: https://arxiv.org/abs/1307.2968

[31] H. Zhao, J. Wang, F. Liu, Q. Wang, N. Luo, and W. Zhang, "Resource allocation for virtual streaming media server cluster in cloud-based multi-version VoD," in *Proc. IEEE 22nd Int. Conf. Comput. Supported Cooperat. Work Design (CSCWD)*, May 2018, pp. 313–318.

[32] J. Choi, A. S. Reaz, and B. Mukherjee, "A survey of user behavior in VoD service and bandwidth-saving multicast streaming schemes," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 1, pp. 156–169, 1st Quart., 2012.
[33] S. M. S. Tanzil, W. Hoiles, and V. Krishnamurthy, "Adaptive scheme for caching YouTube content in a cellular network: Machine learning approach," *IEEE Access*, vol. 5, pp. 5870–5881, 2017.
[34] V. K. Adhikari, S. Jain, Y. Chen, and Z.-L. Zhang, "Vivisecting YouTube: An active measurement study," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 2521–2525.
[35] F. Long and X. Wang, "Predicting user requests in practical VoD systems," in *Proc. IEEE Workshop Adv. Res. Technol. Ind. Appl. (WARTIA)*, Sep. 2014, pp. 1294–1297.
[36] C. Li, J. Liu, and S. Ouyang, "Large-scale user behavior characterization of online video service in cellular network," *IEEE Access*, vol. 4, pp. 3675–3687, 2016.
[37] I. Gradshten, I. Ryzhik, A. Jeffrey, and D. Zwillinger, *Table of Integrals, Series, and Products*. New York, NY, USA: Academic, 2007.
[38] G. P. Patil, "On the evaluation of the negative binomial distribution with examples," *Technometrics*, vol. 2, no. 4, pp. 501–505, Nov. 1960.
[39] C. Forbes, M. Evans, N. Hastings, and B. Peacock, *Statistical Distribution*. Hoboken, NJ, USA: Wiley, 2011.

TIANJIAO WANG received the B.Sc. degree in electronic and communication engineering from Qufu Normal University and the M.Sc. degree in electronic and communication engineering from Beihang University. She is currently pursuing the Ph.D. degree with the Department of Electrical Engineering, City University of Hong Kong. Her research interests are in the areas of popularity prediction and path planning.

CHAMIL JAYASUNDARA received the B.Sc. degree in electrical and electronic engineering from The University of Peradeniya, Sri Lanka, and the Ph.D. degree in electrical and electronic engineering from The University of Melbourne, Australia. From 2009 to 2018, he was with The University of Melbourne. His research interests include energy efficiency, access network architectures, IPTV distribution networks and content management, and applications of machine learning.

MOSHE ZUKERMAN (Life Fellow, IEEE) received the B.Sc. degree in industrial engineering and management and the M.Sc. degree in operations research from the Technion–Israel Institute of Technology, Haifa, Israel, and the Ph.D. degree in engineering from the University of California, Los Angeles, CA, USA, in 1985. He was an independent Consultant with IRI Corporation and a Postdoctoral Fellow with the University of California, from 1985 to 1986. From 1986 to 1997, he was with Telstra Research Laboratories (TRL), as a Research Engineer, and from 1988 to 1997, as a Project Leader. He also taught and supervised graduate students at Monash University, from 1990 to 2001. From 1997 to 2008, he was with The University of Melbourne, Melbourne, VIC, Australia. In 2008, he joined the City University of Hong Kong, as the Chair Professor of information engineering, and as the Team Leader. He has more than 300 publications in scientific journals and conference proceedings. He has served on various editorial boards, such as *Computer Networks*, the *IEEE Communications Magazine*, the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, the IEEE/ACM TRANSACTIONS ON NETWORKING, and the *International Journal of Communication Systems*.

AMPALAVANAPILLAI NIRMALATHAS (Senior Member, IEEE) received the Ph.D. degree in electrical and electronic engineering from The University of Melbourne.

Over the past two decades, he has held many senior leadership positions at The University of Melbourne, including the Head of the Department in electrical and electronic engineering. He has also held Visiting Scientists appointments at NICT Japan and I2R Singapore. He is currently a Professor of electrical and electronic engineering with The University of Melbourne and the Director of the Networked Society Institute–Interdisciplinary Research Institute focusing on challenges and opportunities arising from the society's transition towards a networked society. He has written more than 450 technical articles. His current research interests include energy efficient telecommunications, access networks, optical-wireless network integration, and broadband systems and devices.

ELAINE WONG received the Ph.D. degree from The University of Melbourne, Melbourne, Australia, in 2002. She is currently the Professor and the Associate Dean of the Melbourne School of Engineering, The University of Melbourne. She has coauthored more than 200 journal and conference publications. Her research interests include energy-efficient optical and wireless networks, optical networking in metro-access and fronthaul networks, optical-wireless integration, broadband applications of VCSELs, optical wireless networks, wireless sensor body area networks, and emerging optical and wireless technologies for the Tactile Internet. She is also serving as a TPC Member of OFC and ECOC. She has previously served on the Editorial Boards of the IEEE/OSA JOURNAL OF LIGHTWAVE TECHNOLOGY and the IEEE/OSA JOURNAL OF OPTICAL NETWORKING AND COMMUNICATIONS.

CHATHURIKA RANAWEERA received the Ph.D. degree from The University of Melbourne, Australia, in 2014. She is currently a Senior Lecturer with Deakin University, Melbourne. Her research interests include optical-wireless convergence, optical transport network architectures, the IoT connectivity, energy-efficient communications and networks, network optimization, and machine learning applications.

CHANG XING received the B.Eng. degree in electronic information engineering from the Hebei University of Science and Technology, Hebei, China, and the M.Sc. degree from the City University of Hong Kong, in 2014, where she is currently pursuing the Ph.D. degree with the Department of Electrical Engineering. Her research interests include telecommunication network optimization and multilayered network design.

BILL MORAN (Life Member, IEEE) received the B.Sc. degree (Hons.) in mathematics from the University of Birmingham, in 1965, and the Ph.D. degree in pure mathematics from the University of Sheffield, U.K., in 1968. From 2014 to 2017, he was the Director of the Signal Processing and Sensor Control Group, School of Engineering, RMIT University, from 2001 to 2014, a Professor with the Department of Electrical Engineering, The University of Melbourne, the Director of the Defence Science Institute, The University of Melbourne, from 2011 to 2014, a Professor of mathematics, from 1976 to 1991, the Head of the Department of Pure Mathematics, from 1977 to 1979 and 1984 to 1986, the Dean of the Department of Mathematical and Computer Sciences, in 1981, 1982, and 1989, with The University of Adelaide, and the Head of the Mathematics Discipline, Flinders University of South Australia, from 1991 to 1995. He was the Head of the Medical Signal Processing Program, Cooperative Research Centre for Sensor Signal and Information Processing, from 1995 to 1999. He has been a Professor of defence technology with The University of Melbourne, since 2017. He also works in various areas of mathematics including harmonic analysis, representation theory, and number theory. His main areas of research interest are in signal processing both theoretically and in applications to radar, waveform design and radar theory, sensor networks, and sensor management. He was a member of the Australian Research Council College of Experts, from 2007 to 2009. He was elected to the Fellowship of the Australian Academy of Science, in 1984. He has been a Principal Investigator on numerous research grants and contracts, in areas spanning pure mathematics to radar development, from both Australian and U.S. Research Funding Agencies, including DARPA, AFOSR, AFRL, Australian Research Council (ARC), Australian Department of Education, Science and Training, and Defence Science and Technology, Australia.