Popularity Awareness in *Temporal-DHT* for P2P-based Media Streaming Applications

Abhishek Bhattacharya, Zhenyu Yang and Deng Pan
School of Computing and Information Sciences
Florida International University
Email: {abhat002, yangz, pand}@cis.fiu.edu

***Abstract*—Application-layer overlay networks are receiving considerable popularity due to its flexibility and readily deployable nature thereby providing support for a plethora of Peer-to-Peer (P2P) applications. Currently, the real-world deployment of Internet-scale P2P media streaming systems involve the usage of tracker server for content discovery in on-demand model with asynchronous interactivity. The inherent drawbacks of tracker-server based approach are obvious due to scalability and bottleneck issues, which prompted us to pursue a structured P2P based proposition such as Distributed Hash Tables (DHT) which are already proved to be stable substrates. The challenging issue of accommodating a large number of update operations with the continuous change of user's playing position in DHT-based overlay is addressed in our previous work by the concept of *Temporal-DHT* which exploits the temporal dynamics of the content to estimate playing position. In this paper, we incorporate the notion of *popularity awareness* in the Temporal-DHT framework which will help to adapt the query resolution mechanism by addressing the skewness of content popularity typically found in real multimedia user access patterns. The essential objective of popularity awareness mechanism is to increase the overall performance of Temporal-DHT by optimizing the search cost among the entire content set within the system. We formulate the problem and provide practical solutions with extensive simulation results that demonstrates the effectiveness of popularity-aware Temporal-DHT by achieving optimized query resolution cost and high streaming quality for on-demand systems in a dynamic network environment where user's are free to asynchronously join/leave the system.**

***Index Terms*—peer-to-peer; popularity awareness; streaming; overlay network;**

I. Introduction

P2P revolution started with the initial *unstructured* systems such as Gnutella, Napster, etc. slowly followed by the more efficient *structured* approaches such as DHTs typically represented by Chord [23], CAN [19], Pastry [21], etc. P2P supported applications started with web caching, distributed storage, etc. slowly followed by the more popular ones such as file sharing (e.g., BitTorrent [5]), multicasting (e.g., Narada [12]) and live-streaming (e.g., CoolStreaming [28], PPLive [22], AnySee [15], etc.). The potential advantage of P2P-based applications is mainly associated with the fact that peers share their resources such as processing power, storage and bandwidth to help each other in searching/distributing content, thereby considerably alleviating the server load. Internet traffic is largely dominated by multimedia data which is bandwidth-hungry in nature, therefore imposing more importance to the management and distribution of content particularly critical with respect to P2P applications.

Some of the recent studies revealed that On-demand streaming can be immensely benefited from the application of P2P techniques [13], [14]. To address the challenging problem of efficient content discovery in On-demand systems, we advocate a DHT-overlay based approach which is already proved to be stable a substrate with nice characteristics such as scalable, decentralized control, self-organizing, and resilient to network/peer dynamics. Incorporating DHT in On-demand streaming systems is not a trivial issue since it will generate a flurry of update operations with the continuously changing play position of the user. This difficult issue of accommodating a large number of update operations is addressed by the concept of *Temporal-DHT* [1] which exploits the temporal dynamics of the content to estimate playing position. Temporal-DHT combines the advantage of high streaming efficiency due to the buffer overlap relation between parent and child peers from *cache-and-relay* based approach with more adept support in dynamic and asynchronous operations such as random jumps by avoiding the dependency on playing position between peers typically represented through *static-cache* based approaches. Temporal-DHT employs a skilful integration of static and dynamic buffer management schemes to handle the request dynamics and streaming efficiency in a seamless fashion. It is an augmented version of generic DHT semantics by implementing *query reformulation*, *TTL filtering*, and *access workload self-profiling* techniques.

Popularity Awareness concept is generally employed for optimizing the search cost of the system by exploiting the content popularity factor. The primary intention is to reduce the search cost of more popular contents since they are queried frequently which will ultimately help to improve the overall performance of the system [18]. It is already found to be useful in web caching and file sharing systems where the data objects are typically characterized with varying popularity ratios. Web requests in the Internet are found to be highly skewed with a Zipf-like distribution [26] with typical characteristics of

978-0-7695-4589-9 2011

DOI 10.1109/ISM.2011.46

a few objects having a very high popularity, a medium number of objects with average popularity, followed by a huge number of objects with very low popularity. Zipf-distributions are universally used for popularity modeling in various scenarios. It is usually realized by replicating the data objects at the various intermediate nodes in the query resolution path which eventually helps to reduce the number of search hops of the popular contents. The process should be adaptive under varying popularity scenarios since there is an associated tradeoff relation between the higher performance due to lower search complexity and the replication cost for caching the data objects at various intermediate nodes. The replication context is not applicable for media applications since it does not make sense to continuously cache large-sized media objects which consumes a lot of network bandwidth. VMesh [26] employs a popularity-based segment storage mechanism where the cached segments are continuously replaced in accordance with the current content popularity distribution. The downside of this mechanism is the network bandwidth consumed for constantly replacing the media objects which makes it a *heavy-weighted* technique.

The incorporation of popularity awareness in Temporal-DHT is performed in a unique context with a *light-weighted* fashion by adapting the update interval based on the popularity distribution. Temporal-DHT employs a query reformulation technique where the generic exact match DHT prefix routing is augmented with a range query and the span of the range query is dependent on the object update interval. The current Temporal-DHT framework assumes a fixed value for the object update interval thereby rendering increased search cost with respect to popularity skewness of content. There exists a tradeoff relation between the performance benefits of decreasing search cost and the increased cost of update operations i.e., if we intend to minimize the search cost then we need to decrease the update interval which will trigger more number of update operations thereby increasing the messaging overhead. Due to this situation, it is essential to find an efficient solution to optimize the search-update cost in the context of popularity-awareness, whereby each data objects will have different update intervals based on the popularity distribution. We address the following important problem: *How to minimize the search cost with a given threshold constraint of update interval?* In this paper, we propose a technique to adapt the update interval for optimizing the search cost according to the varying popularity distribution of the content in the context of a Temporal-DHT with the primary objective of reducing the search cost of the popular data objects.

To summarize, our contributions in this paper are as follows: (a) We incorporated the notion of popularity awareness in the context of a Temporal-DHT with the objective of optimizing the search and update cost in varying conditions, (b) We formulate the problem in a representative manner and propose solutions to achieve the objective, (c) We implement the solutions in a Temporal-DHT based P2P Video-on-Demand system model and provide extensive simulation studies to show the effectiveness of the popularity awareness approach in a media streaming scenario and the performance benefits associated with the optimization of search cost. The rest of this paper is organized as follows: We present some basic background stuff related to DHT and Temporal-DHT in Sec. II which will be required to understand the problem. Sec. III illustrates the detailed functional mechanism of the popularity awareness approach and its interpretation in Temporal-DHT systems. We analyze our experimental results from our simulation study in Sec. IV. We summarize related work from the literature in Sec. V. Finally, we conclude the paper in Sec. VI.

II. Background

We provide some basic and specific details of Temporal-DHT and P2P VoD systems which will be required to understand the later parts and also develop terminologies/notations represented in Table I for later reference.

TABLE I: List of used symbols

Definition	Notation
Participating Peers	p_i with $P = \{p_1, p_2, .., p_N\}$ of size $N = \|P\|$
In/Out-bound Bandwidth	I_i/O_i of p_i
Video Server	S with out-bandwidth S_o
Video Stream	$C = \{c_1, c_2, .., c_M\}$ of size $M = \|C\|$
Video Segment Size	D MB
Video Data Rate	d Kbps
Play time of one video segment	$t_M = \frac{D*1000*8}{d}$
Dynamic/Random Buffer	B_p of size k segments (i.e., kD MB)
Static/Sequential Buffer	B_s of size b segments (i.e., bD MB)
Publish Interval	$T = z \times t_M$
Time-To-Live	TTL
Content Successor/Predecesor pointers	p_{succ}/p_{pred}

Temporal-DHT is a novel conceptual augmentation to the traditional DHT for indexing content with temporal dynamics which provides considerable savings in messaging overhead as already proposed in our earlier work [1]. It cherishes two distinctive properties: (1) *Application-level Characteristics*: Temporal-DHT advocates a new DHT-proactive design approach as we believe that DHT can provide better service if it knows the internal behavior of the application and take a more active role; (2) *Data Transiency*: Temporal-DHT exploits predictive temporal dynamics of the content for effective query resolution and the indexing records are flushed off from the system at a periodic interval.

A typical Temporal-DHT indexing record is a tuple of $\langle p, c, TTL \rangle$ with $p \in P$ and $c \in C$, which indicates that

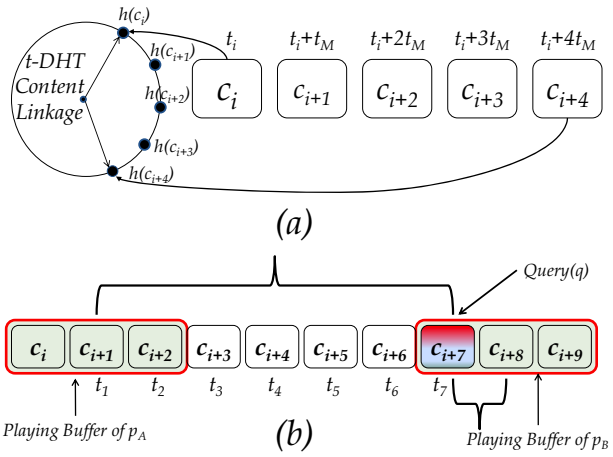

Fig. 1: Sketch to illustrate (a) *Temporal-DHT* content linkage and updates and (b) range query reformulation and buffer sliding

peer p holds segment c and TTL (Time-To-Live) specifies the expiration time of the record. For the indexing of dynamic segments, TTL is initialized to z whereas for the indexing of static segments it is initialized to ∞. Temporal-DHT performs *lazy updations* i.e., indexing records are updated in a predefined constant periodic interval T and therefore employing *query reformulation* and *TTL filtering* techniques to improve query accuracy by taking hint from the dynamics of content workload and allowing certain degree of inconsistencies in the DHT indexing structure. We present an intuitive illustration of the Temporal-DHT indexing and range query as follows: Referring to Fig: 1(a) and assuming k=1, we can observe that a *Temporal-DHT* update $\langle p_i, c_i, 4 * t_M \rangle$ is performed by VoD peer, p_i at t_i with z=4. The buffer slides by one segment after each t_M interval and the next *Temporal-DHT* update $\langle p_i, c_{i+4}, 4 * t_M \rangle$ is performed at t_i+4∗t_M by p_i. Any *Temporal-DHT* query, q posted by other VoD peers for c_{i+4} during time interval $[t_i, t_i + 4t_M - \delta]$ (i.e. δ is a very small time unit which signifies c_{i+4} already loaded in B_p but the *Temporal-DHT* update is not yet performed) needs to be transformed into a range query $\langle q, q - z \rangle$ for effectively reaching to p_i. The accurate query resolution phenomenon is formalized in Theorem II.1 taken from [1]. For proofs and other *TTL filtering* details, please refer to [1].

Theorem II.1. *Given the size of playback buffer k and the publish interval z, a peer that searches for dynamic segment c_i needs to perform a range query of at most $k + z$ segments.*

Temporal-DHT initiates a content-based overlay for accelerating the range query process where the linkage pointers represent a semantic sequential relationship between segments, such that $c_1 \leftrightarrow c_2 \leftrightarrow ... \leftrightarrow c_M$. The content linkage is used to facilitate fast in-order access. The content distance is used to decide whether a query should be forwarded through content linkage or generic DHT routing; the idea is similar to VMesh [26]. To harness the advantage of both static-cache and cache-relay approaches, a *Temporal-DHT based Mesh* (TDHTM) was designed which provides an overarching framework that seamlessly integrates the power of asynchronous interactivity support from static indexing and smooth streaming efficiency from dynamic indexing. TDHTM administers a combined static-dynamic buffer management which is basically a combination of B_p and B_s wherein the B_s segments are indexed with $TTL = \infty$ kept constant throughout the peer's lifetime followed by the B_P segments are published to the Temporal-DHT with $TTL = z$ and performs *i.e. buffer sliding* after each segment playback. Dynamic indexing involves the publication of indexing record in a periodic interval of T and any query is reformulated by a range search whereas static indexing involves a one-time publication of indexing record at initialization and its query is processed as a generic DHT routing based resolution. TDHTM involves *adaptive content distribution* by adaptively switching between random seek mode (handled by static indexing) and continuous playback mode (handled by dynamic indexing) with the help of *access workload self-profiling* at the client end.

III. Details

In this section we present our detailed strategy for incorporating popularity-awareness in the context of a Temporal-DHT based P2P VoD system model.

A. Search Cost

In this section we will analyze the cost of a search query in Temporal-DHT by considering the range query reformulation technique as already described before. A typical Temporal-DHT search query consist of two parts: (a) an initial exact match generic DHT query resolution with prefix routing through the help of finger table, followed by (b) a Temporal-DHT range query resolution by linear traversal of the content-based overlay through the p_{succ}/p_{pred} linkage pointers in the forward/backward directions respectively. From [23], based on Chord's finger allocation principle, the query cost between any pair of source and destination pair is given by $O(logN)$. This can be derived from the fact that in each routing hop it is able to traverse at most half of the remaining distance between the source and the destination in the identifier space i.e., let the ith. node in Chord have a node ID i, then the kth. item in the finger table points to the successor node of ID $(i + 2^{k-1})$ where $1 \leq k \leq logN$ and therefore, the distance travelled by a routing hop is given by 2^x for $1 \leq x \leq logN$ and the query is forwarded to the node in the xth. entry from the finger table. Next, the search cost for the range query part of Temporal-DHT can be derived from Theorem: II.1 where it is stated that the maximum search span consist of

$k + z$ segments. Each segment can be probed by a single message hop and therefore the total complexity for the range search can be equated to $O(k + z)$. Thus, the total search cost in terms of number of hops to search any content object c_x is given as follows:

$$H_x = O(logN + k + z) \tag{1}$$

This result can be equivalently mapped to the result in Theorem: 4 from [18] with a slight change since their work is based in the context of replication.

B. Problem Formulation

We propose the following problem in the context of Temporal-DHT:

- ***MIN-SEARCH***: Minimize the total query cost (H) in terms of lookup hops with a given threshold constraint of update interval ($z_{threshold}$).

It can be observed that this problem is relevant in the context of a Temporal-DHT for maximizing the performance benefits with respect to the messaging complexity of the system which consumes valuable network bandwidth. This problem is ignored in the previous Temporal-DHT framework [1] by providing a constant value of z thereby increasing the cost of the overall system which will be efficiently addressed in this paper.

C. MIN-SEARCH

Based on the definition of popularity, content c_x with popularity quotient p_x receives a fraction p_x of all searches. The average search cost (H_x) was derived in Eq: (1). Given certain content popularity distribution, the total search cost H of M data objects (i.e., video stream divided into M number of segments) can be represented as follows:

$$H = \sum_{x=1}^{M} (p_x \cdot H_x) = \sum_{x=1}^{M} p_x \cdot (logN + k + z) \tag{2}$$

The optimization objective is to minimize the value of H. For the above problem, we derive our solutions from Theorem: 5 in [18] by modifying it to fit in our scenario since our problem can be equivalently mapped to the problem in [18]. It is stated as follows:

- *Let the cost of each update operation be denoted as l_x (i.e., $l_x = \frac{1}{z}$) and the total number of update operations as L, then it is observed that for $\sum_{x=1}^{M} l_x = L = \frac{M}{z}$, H is minimized when $\forall x : \; l_x = p_x \cdot L = p_x \cdot \frac{M}{z}$.*

In accordance to the popularity-based proportional principle, we verify by substituting $l_x = p_x \cdot L$ into Eq: (2) and get:

$$H = logN + k - logL - \frac{M}{z} \sum_{x=1}^{M} (p_x \cdot logp_x) \tag{3}$$

We notice that the term $\sum_{x=1}^{M} p_x \cdot logp_x$ is actually the entropy of the popularities p_x. This is in concurrence with the intuition since we have expected that popularity distribution skewness will play a major role in the cost optimization objective, and taking the popularity as an entropy is a sound measure of skew of the distribution. Thus, we can observe that the average search cost H depends upon N, k, L, M, z and the entropy of p_x.

D. Estimation of Content Popularity

We employ a practical approach for estimating popularity with the help of distributed averaging algorithm in a decentralized fashion. We exploit the algorithm proposed in [26] to calculate the average number of segment request received over distributed nodes for estimating the content popularities. A brief description of the algorithm is as follows: Each node connects to r random neighbors and exchange messages with them. Assume node i has a local value of z_i and the requirement is to estimate the average value of all z_i over the network. Additionally, each node maintains a local dynamic variable x_i which is initialized with a value of z_i. Each node periodically communicates with its neighbors and performs a set of actions as follows: (a) Node i sends its local value x_i to Node j; (b) Node j updates its local value x_j to $x_j + \gamma_j(x_i - x_j)$ where $0 < \gamma_j < 1$ is a local parameter. Node j also send back a value $\gamma_j(x_i - x_j)$ to node i; (c) Node i updates its local value x_i to x_j to $x_j - \gamma_j(x_i - x_j)$. The central idea behind the algorithm is to conserve the sum of all the values in the system by performing alternative increment and decrement operation between two neighboring nodes thereby approaching closer to the average value at each update. The algorithm can be extended to cope with node dynamics where each node i maintains a variable η_{ij} associated with each neighbor j which aggregates all the changes made due to j. On detection of node j's departure, node i substract η_{ij} from x_i which essentially ensures the conservation of the total sum of values.

The above algorithmic technique is utilized to keep track of the total number of each segment requests from different users in the context of Temporal-DHT which will be used to estimate the popularity quotients. Each Temporal-DHT peer maintains an array a_i for each segment i indicating its access to segment i. If the peer receives a request for segment i, then it sets $a_i = 1$ otherwise $a_i = 0$. A Temporal-DHT peer also maintains a local set of variables b_i which keeps track of the number of received requests of segment i and runs the averaging algorithm to exchange and update b_i continuously with its neighboring nodes. The information gets disseminated through each peers neighborhood to eventually arrive on a local b_i value which represents a good approximation of the global popularity distribution for every segment i. It is now trivial to compute the estimated popularity $\hat{p_i}$ of segment i from its local set of average b_i values as follows:

$$\hat{p_i} = \frac{b_i}{\sum_{x=1}^{M} b_x} \tag{4}$$

Thus, we are able to estimate popularity $\hat{p}_i$ distribution coefficients which will help to improve performance by proportionately adapting the update interval separately for each segment with the help of the techniques discussed in Sec: III-C.

IV. Experiments

In this section we present our simulation results for Popularity-Aware Temporal-DHT based Mesh (TDHTM-PA) by studying different system properties. The simulator is implemented in C++, featuring a discrete event-driven timeout mechanism for various P2P operations. All peer dynamics, including random jumps, joins and departures, are simulated with events scheduled at their respective times. Chord [23] is used as the base DHT overlay due to its simplistic construction and provable performance guarantees. To portray the distinctive features of TDHTM-PA, we have also implemented VMesh [26] for experimental comparison. We also experimentally compared TDHTM-PA with Temporal-DHT based Mesh (TDHTM) which is the same framework without the popularity-awareness module. This comparison will essentially draw out the performance benefits of the popularity-awareness concept if any found.

A. Simulation Model

We start our discussion with details of network topology followed by system data model.

Network Topology: The underlying physical network topology is generated using GT-ITM [27].The whole network consists of 15 transit domains, each with 25 transit nodes and each transit node is connected to 10 stub domains, each with 15 stub nodes. We randomly place the video server on a transit node and client peers on the stub nodes. The delay of each link is selected proportional to the Euclidean distance between the nodes. For each experimental result presented, we repeat the placement and simulation 10 times to mitigate the effect of randomness.

Data Model: The TDHTM-PA overlay size or *user population* (i.e., N) is varied in the range of 256 to 4096. We model the user arrival process as a Poisson distribution with an average inter-arrival time λ = 1 second. The peer lifetime follows an exponential distribution with an expected mean of 30 minutes. The outgoing bandwidth (i.e., O_i) is randomly distributed between 500~1000 Kbps so that it can support a minimum of one stream and a maximum of two streams as the video data rate is d = 500 Kbps. The user request pattern is divided into 2 classes: (a) 50% of the nodes follow a Zipf distribution with various values of α. (b) The rest 50% of the nodes are simulated with a playback workload as hinted by [14] which states that each user initially performs 6~7 random forward/backward jumps on average for scanning the entire video and then settles down with continuous in-order playback. For simplicity, the user viewing lifetime in our experiment is divided into two regions as follows.

- Region 1: The number of random forward/backward jumps is uniformly distributed in the range of 1st~10th minute in each peer's respective life cycle.
- Region 2: The user stabilizes in continuous playback mode starting at the point when Region 1 ends and remains in the same mode till the end of its viewing period.

Each segment size (i.e., D) is set to be 3.84 MB which corresponds to one minute video length. The total viewing length of the video stream is 128 minutes (i.e., M = 128). Each simulation session length is set to be 2 hours. Other parameters are: I_i = 4 Mbps, S_o = 500 Mbps, k = 5, b = 4, and z = 10.

B. Server Stress

We define *server stress* as the number of streams directly supported by the server, which is the average upload bandwidth used by server for supporting the system divided by the data rate of the video stream. Figure 2 plots the server stress as the function of user population. As shown, TDHTM performs much better than VMesh when the system size increases. Specifically for 4096 nodes, the server stress of VMesh is around 998.6 in comparison to 557.6 of TDHTM. TDHTM-PA is found to consistently fare better than TDHTM and the best performance is derived from TDHTM-PA (α=2.0).

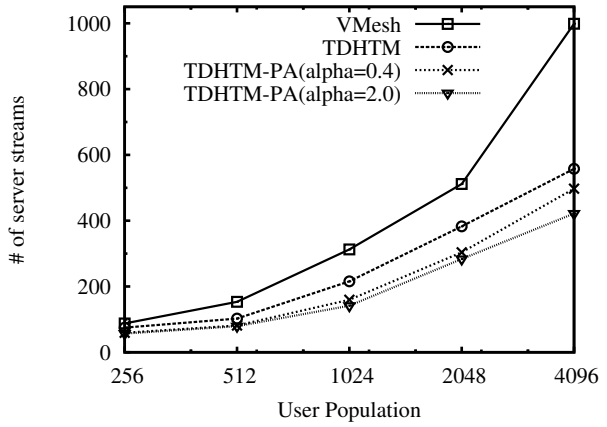

Fig. 2: Server Stress for various user populations.

The reduction of server stress indicates the effectiveness of the integration of the static-dynamic caching technique employed by TDHTM. In VMesh, many peers can request a particular segment from the static buffer of a peer due to the proximity of playing position and since it has limited upload bandwidth, thereby most of the queries may not be resolved resulting in the increase of server stress. The access of segments in TDHTM is better organized. We can infer that popularity-awareness approach provides a substantial amount of performance enhancement on top of a temporal-DHT framework

with different values of Zipf parameter α. The better performance of TDHTM-PA can be reasoned by the fact that it is able to optimize the search cost by an adaptive updation mechanism taking the popularity distribution into account.

C. Streaming Quality

One important streaming quality metric of a video streaming system is *playback continuity* which has significant influence in the user experience. We define playback continuity as the number of segments that are received within deadline divided by the total number of segments a peer should play during its lifetime in the continuous playback mode. Figure 3 plots the playback continuity against various user populations for VMesh, TDHTM and TDHTM-PA. As shown, all the schemes perform pretty good (average above 80%). To further compare, TDHTM performs consistently better than VMesh with around 10% improvement and less variation. If we want to verify the usefulness of popularity-awareness, then we can observe that TDHTM-PA performs better than TDHTM i.e., For a 4096 node network, the continuity index for TDHTM-PA (α=0.4) and TDHTM-PA (α=2.0) are 0.9738 and 0.9837 respectively which is substantially better than 0.9453 of TDHTM. The reason could be attributed to the fact, that optimized search-update cost according to content popularity essentially improves the query success rate especially for the more popular contents which in turn helps to boost the playback continuity.

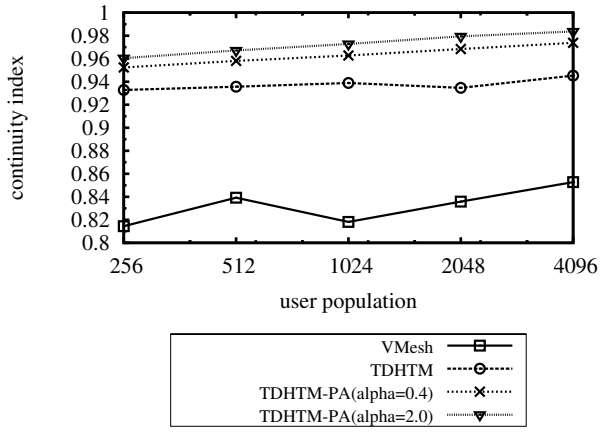

Fig. 3: Playback continuity against various user populations.

Next, we evaluate the distribution of playback continuity in a VoD streaming session which is also important for ensuring fairness and uniform quality dissemination. In case of resource shortage, an ideal P2P-VoD system should be able to gracefully distribute the quality drop among all the participants. Figure 4 shows the distribution of playback continuity with the user population of 1024. We can observe that all the peers achieve a playback continuity in the range of 0.9 ~ 1.0 with an average of 0.9728 which is highly desirable for TDHTM-PA. For TDHTM, the continuity index values span in the range of 0.8 ~ 1.0 and still poorer for VMesh, which confirms the effectiveness of popularity-aware approach by providing high playback continuity to all the peers in the system.

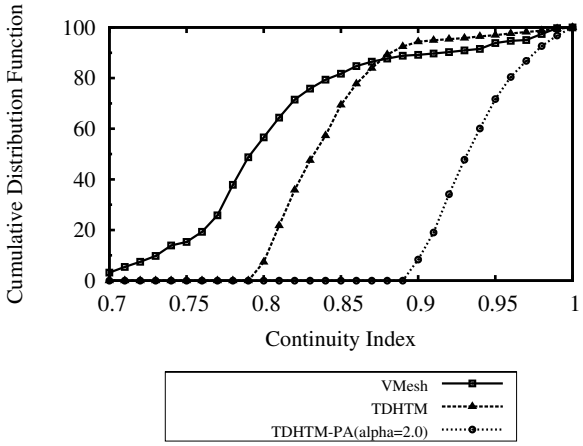

Fig. 4: Distribution of playback continuity (N = 1024).

D. Messaging Overhead

For messaging overhead, we account for only control messages that are required for query/reply with respect to content access and we do not consider initialization/transmission related messages since they are common for both VMesh and TDHTM. Figure 5 plots the messaging overhead in terms of messages/second for different user populations which shows that TDHTM requires lower messaging overhead than VMesh. For example, in a system with 4096 peers, the average messaging cost of VMesh is 148.1 messages/second whereas TDHTM requires 101.5 messages/second, an improvement of 31.5%. Comparing TDHTM with TDHTM-PA, it can be clearly viewed that popularity-based approach provide benefits of lower messaging overhead to a certain extent. The reason is due to the fact that the MIN-SEARCH is involved with minimizing the query cost for a certain constant threshold of update operations thereby reducing overall messaging overhead since search is more expensive than updates.

E. Seek Latency

We define *seek latency* as the time spent acquiring the next segment for playback. Figure 6 plots the average seek latency against various user populations. Our results of VMesh are consistent with those in [26]. The huge difference between the seek latency of TDHTM and VMesh is mainly due to the averaging: in VMesh seeking is needed for almost every segment while in TDHTM it is not needed once a peer joins the multicast overlay tree. Note, the seek latency does not directly

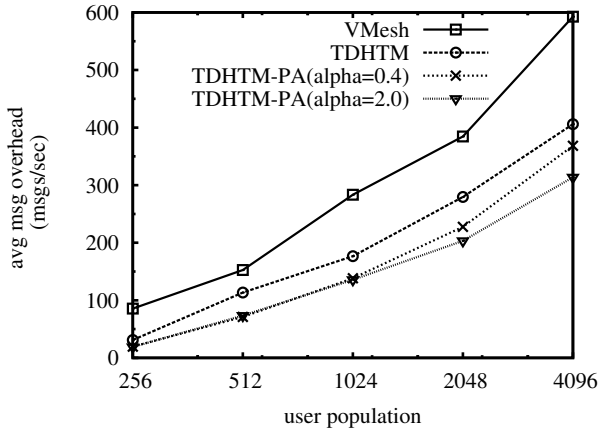

Fig. 5: Messaging overhead against various user populations.

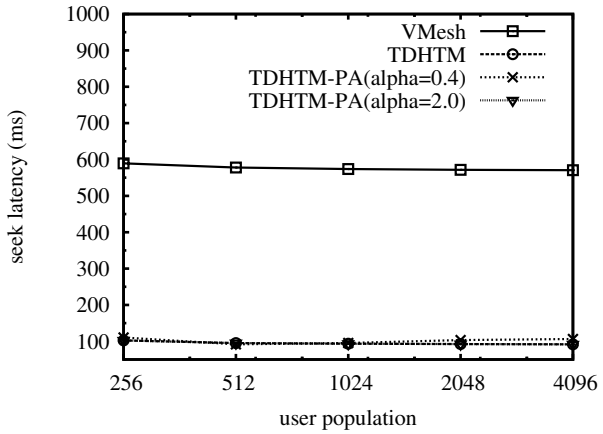

Fig. 6: Seek latency against various user populations.

translate to user experience since the seeking of the next segment is started a little earlier before the end of the current segment. However, a large seek latency may nevertheless increase the chance of missing deadline which impacts the performance of playback continuity as presented earlier. Popularity-awareness fails to induce any effect on the seek latency since the experimental results of the TDHTM and TDHTM-PA are within a very narrow performance range and does not indicate any meaningful deductions.

V. Related Work

In this section, we review related work from two aspects: popularity aware applications and P2P-VoD/DHT based systems.

A. Popularity-aware Applications

Most of the previous work involving popularity-awareness in designing systems have been from the replication point of context. A typical problem in this domain mainly involves in the placement strategies of replicas or cached objects to reduce the search cost for more popular content. Web-caching systems are benefitted from these techniques since the web-based objects follow a Zipf like popularity distribution. CFS [9] is a cooperative file system over Chord [23] which caches the popular objects along the lookup path toward the home nodes where popular objects are originally stored. PAST [20] is a storage system over Pastry [21] where the search for some object is redirected to the nearest replicas of the targeted object. Based on the replication level l, Beehive [17] replicates the object copies to all nodes that have at least l common prefixes matching with the object hash ID. [6] proposed to optimize search efficiency by replication, where the number of replicas of an object is proportional to the square root of the object popularity. [7] presented a square root topology for unstructured P2P networks where the degree of a peer is proportional to the square root of the node popularity. PCache, PRing [18] presented a novel replica placement strategy for web caching systems with data objects having skewed popularities in both deterministic and randomized structured P2P networks. It gave detailed analytical results with closed form optical solutions for different resource optimization objectives. VMesh [26] proposed an interesting angle of incorporating popularity-awareness property in static-cache based DHT overlays in the context of P2P VoD streaming. The distributed static cache is continuously refreshed with different video segments and this segment selection process is based on a probability that has been derived from the segment popularities. Our approach of applying popularity-awareness for optimizing the search-update cost in the context of a Temporal-DHT for a P2P VoD system is unique and different from the above proposals.

B. DHT/P2P-VoD based Systems

In DSL [25], a dynamic skip-list based overlay is proposed where all peers are connected in the base layer according to their playback position in the stream and also randomly attach to a few neighbors in higher layers of the skip-list. oStream [8] utilizes dynamic caching by developing a temporal dependency model among peers and a media distribution tree is computed by the central server to minimize the overall transmission cost. A ring-assisted overlay is deployed in [4] where each peers maintains neighbor based on their similarity of cached content proportional to the radii of the different concentric rings. InstantLeap [16] constructs a hierarchical mesh overlay by dividing all the peers into groups according to playing position. Peers in one group maintain limited membership information of all the other groups through random messaging which helps to support efficient lookup for any random seek operation. PROP [11] uses a distributed storage scheme but relies on proxy servers and global information for cache replacement which are difficult to implement in large scale systems. P2VoD [10] organizes nodes into multi-level clusters according to

their joining times. SplitStream [2] is a high bandwidth content dissemination mesh that is built on top of Pastry [21] and Scribe [3] for multimedia applications. SplitStream constructs multiple Scribe trees based on the internal node disjoint principle and then disseminates each multiple-desription coded (MDC) stream through a separate tree thereby achieving improved robustness against churn. BulletMedia [24] constructs a high-bandwidth overlay mesh by self-organizing the participating peers in a decentralized fashion and also exploys a caching technique by strategically distributing data at different points such that any data object is equally likely to appear at any node. As already discussed, VMesh[26] is a P2P-VoD system with a closely related approach where it performs popularity awareness by replacing the video segments stored in the static cache of each peer proportional to the popularity index and thereby wasting large bandwidth resource.

VI. Conclusion

We incorporated the notion of *popularity awareness* in the Temporal-DHT framework which will help to adapt the query resolution mechanism by addressing the skewness of content popularity typically found in real multimedia user access patterns. The essential objective of popularity awareness mechanism was to increase the overall performance of Temporal-DHT by optimizing the search cost among the entire content set within the system. The basic mechanism involves the adaptation of update operations for minimizing the search cost of popular video segments i.e., the more popular segments increase the updation rate by reducing the value of T (update interval) to lower the search cost since a Temporal-DHT query cost is dependent on the update interval. We formulated the problem and provided practical solutions with extensive simulation results that demonstrate the effectiveness of popularity-aware Temporal-DHT.

References

[1] A. Bhattacharya, Z. Yang, and S. Zhang. Temporal-DHT and Its Application in P2P-VoD Systems. In *ISM '10*, pages 81–88, dec. 2010.

[2] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. Splitstream: high-bandwidth multicast in cooperative environments. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 298–313, New York, NY, USA, 2003. ACM.

[3] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron. Scribe: a large-scale and decentralized application-level multicast infrastructure. *Selected Areas in Communications, IEEE Journal on*, 20(8):1489–1499, oct 2002.

[4] B. Cheng, H. Jin, and X. Liao. Supporting vcr functions in p2p vod services using ring-assisted overlays. In *ICC '07*, pages 1698 –1703, june 2007.

[5] B. Cohen. Incentives Build Robustness in BitTorrent. *In Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems*, 2003.

[6] E. Cohen and S. Shenker. Replication strategies in unstructured peer-to-peer networks. In *SIGCOMM '02*, SIGCOMM '02, pages 177–190, New York, NY, USA, 2002. ACM.

[7] B. F. Cooper. An optimal overlay topology for routing peer-to-peer searches. In *Middleware '05*, Middleware '05, pages 82–101, New York, NY, USA, 2005. Springer-Verlag New York, Inc.

[8] Y. Cui, B. Li, and K. Nahrstedt. ostream: asynchronous streaming multicast in application-layer overlay networks. *Selected Areas in Communications, IEEE Journal on*, 22(1):91 – 106, jan. 2004.

[9] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with cfs. In *SOSP '01*, SOSP '01, pages 202–215, New York, NY, USA, 2001. ACM.

[10] T. Do, K. Hua, and M. Tantaoui. P2VoD: providing fault tolerant video-on-demand streaming in peer-to-peer environment. In *Communications, 2004 IEEE International Conference on*, volume 3, pages 1467 – 1472 Vol.3, june 2004.

[11] L. Guo, S. Chen, and X. Zhang. Design and evaluation of a scalable and reliable p2p assisted proxy for on-demand streaming media delivery. *Knowledge and Data Engineering, IEEE Transactions on*, 18(5):669 – 682, may 2006.

[12] Y. hua Chu, S. Rao, S. Seshan, and H. Zhang. A case for end system multicast. *Selected Areas in Communications, IEEE Journal on*, 20(8):1456 – 1471, oct 2002.

[13] C. Huang, J. Li, and K. W. Ross. Can internet video-on-demand be profitable? In *SIGCOMM '07*, New York, NY, USA, 2007.

[14] Y. Huang, T. Z. Fu, D.-M. Chiu, J. C. Lui, and C. Huang. Challenges, design and analysis of a large-scale p2p-vod system. In *SIGCOMM '08*, pages 375–388, New York, NY, USA, 2008. ACM.

[15] X. Liao, H. Jin, Y. Liu, L. M. Ni, and D. Deng. Anysee: Peer-to-peer live streaming. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1 –10, april 2006.

[16] X. Qiu, C. Wu, X. Lin, and F. C. Lau. Instantleap: fast neighbor discovery in p2p vod streaming. In *NOSSDAV '09*, pages 19–24, New York, NY, USA, 2009. ACM.

[17] V. Ramasubramanian and E. G. Sirer. The design and implementation of a next generation name service for the internet. In *SIGCOMM '04*, SIGCOMM '04, pages 331–342, New York, NY, USA, 2004. ACM.

[18] W. Rao, L. Chen, A.-C. Fu, and G. Wang. Optimal resource placement in structured peer-to-peer networks. *Parallel and Distributed Systems, IEEE Transactions on*, 21(7):1011 –1026, july 2010.

[19] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *SIGCOMM '01*, SIGCOMM '01, pages 161–172, New York, NY, USA, 2001. ACM.

[20] A. Rowstron and P. Druschel. Storage management and caching in past, a large-scale, persistent peer-to-peer storage utility. In *SOSP '01*, SOSP '01, pages 188–201, New York, NY, USA, 2001. ACM.

[21] A. I. T. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Middleware '01*, Middleware '01, pages 329–350, London, UK, 2001. Springer-Verlag.

[22] S. Spoto, R. Gaeta, M. Grangetto, and M. Sereno. Analysis of pplive through active and passive measurements. In *Parallel Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on*, pages 1 –7, may 2009.

[23] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM '01*, pages 149–160, New York, NY, USA, 2001. ACM.

[24] N. Vratonjić, P. Gupta, N. Knežević, D. Kostić, and A. Rowstron. Enabling dvd-like features in p2p video-on-demand systems. In *Proceedings of the 2007 workshop on Peer-to-peer streaming and IP-TV*, P2P-TV '07, pages 329–334, New York, NY, USA, 2007. ACM.

[25] D. Wang and J. Liu. A dynamic skip list-based overlay for on-demand media streaming with vcr interactions. *Parallel and Distributed Systems, IEEE Transactions on*, 19(4):503 –514, april 2008.

[26] W.-P. Yiu, X. Jin, and S.-H. Chan. Vmesh: Distributed segment storage for peer-to-peer interactive video streaming. *Selected Areas in Communications, IEEE Journal on*, 25(9):1717 –1731, december 2007.

[27] E. Zegura, K. Calvert, and S. Bhattacharjee. How to model an internetwork. In *INFOCOM '96*, volume 2, pages 594 –602 vol.2, 1996.

[28] X. Zhang, J. Liu, B. Li, and Y.-S. Yum. Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 3, pages 2102 – 2111 vol. 3, march 2005.