# Dynamic Adaptive Streaming over Popularity-driven Caching in Information-Centric Networks

Wenjie Li, Sharief M.A. Oteafy, Hossam S. Hassanein
School of Computing
Queen's University, Kingston, ON, Canada
Email: {liwenjie, oteafy, hossam}@cs.queensu.ca

***Abstract*—The growing demand for video streaming is straining the current Internet, and mandating a novel approach to future Internet paradigms. The advent of Information-Centric Networks (ICN) promises a novel architecture for addressing this exponential growth in data-intensive services, of which video streaming is projected to dominate (in traffic size). In this paper, we present a novel strategy in ICNs for adaptive caching of variable video contents tailored to different sizes and bit rates. Our objective is to achieve optimal video caching to reduce access time for the maximal requested bit rate for every user. At its core, our approach capitalizes on a rigorous delay analysis and potentiates maximal serviceability for each user. We incorporate predictors for requested video objects based on a popularity index (Zipf distribution). In our proposed model, named *DASCache*, we present delay queuing analysis for cached objects, providing a cap on expected delay in accessing video content. In *DASCache*, we present a Binary Integer Programming (BIP) formulation for the cache assignment problem, which operates in rounds based on changes in content requests and popularity scores. *DASCache* reacts to changes in network dynamics that impact bit rate choices by heterogeneous users and enables users to stream videos, maximizing Quality of Experience (QoE). To evaluate the performance of *DASCache*, in contrast to current benchmarks in video caching, we present an elaborate performance evaluation carried out on ndnSIM, over NS-3.**

***Keywords*—Dynamic adaptive streaming; Content-centric network; in-network caching; Optimization; Popularity index; Queueing analysis; Binary integer programming.**

## I. Introduction

The dynamics of the Internet, which have scaled superbly over the past twenty years, are currently faltering under the projected growth of data demands. While several attempts at improving network performance via Content Distribution Networks (CDNs) and web proxy caching have held their ground for years, there is significant concern as to their ability to cope with an exponential growth in data. More prominently, multimedia data is expected to increase and amount to a considerable portion of Internet traffic. It is projected that, by 2016, video will amount for almost 55% of overall data traffic [1] in conservative estimates.

In recent years, a rising concern in the paradigms governing the Internet has resulted in the emergence of Information-Centric Networks (ICN). They are proposed as the next generation of the Internet designed to intrinsically handle large content and distribute it via network layer primitives. In ICNs, the core premise is adapting to content and catering to both consumers and producers, rather than adopting the client-server approach of the current host-centric Internet. Where CDNs and peer to peer (P2P) networks provided overlays over the Internet to handle large content dissemination, ICN exploits caching as a networking primitive to enhance content delivery. Currently, several ongoing research projects are implementing versions of the ICN vision. Pioneering among these architectures is the Content-Centric Network (CCN) developed by Jacobson *et al.* in [2]. CCN stands out as one of the prominent architectures that witnessed significant uptake by the research community; facilitating rapid benchmarking and insightful performance analysis across proposed protocols and schemes.

The challenge of catering to video content, as a dominant type of traffic, is instrumental in the development of ICNs. To this end, some researchers attempted to produce caching strategies that would improve cache utilization, such as transcoding. Yet, the problem of addressing heterogeneous caching over variable chunks of different bit rates, without incurring in-network data processing, stands unsolved. In this paper, we present a novel model, *DASCache*, to handle heterogeneous video content caching in ICNs. Our objective is simply to minimize the average *access time per bit* for requested video content, to facilitate rapid streaming over ICNs which improves Quality of Experience (QoE) for all users. Our approach addresses variable bit rates and content sizes, to mimic realistic scenarios where different users would demand the best-possible bit rate given the heterogeneity of their devices and link conditions. Utilizing *DASCache*, users with varied network conditions will experience higher throughput and are thus able to switch to videos with better resolution, achieving the best experience possible.

The remainder of the paper is organized as follows. Section II overviews related work including recent research on caching schemes in ICNs and predecessor work on dynamic adaptive streaming. Section III presents our *DASCache* cache management strategy, encompassing the queueing model to derive expected caps on access delay. We present the rigorous formulation of *DASCache* in Section IV. Our experiment setup and performance evaluation results are detailed in Section V and we conclude in Section VI with our final remarks and propositions for future work in this viable direction.

## II. Related Work

In the realm of ICNs, content could be cached when travelling towards the requester in order to provide flexibility via on-path caching mechanisms. Recent research efforts have been devoted to the cache decision policy problem over CCN and the solution could be generally divided into *explicit* or *implicit*, according to the degree of coordination. The explicit caching policy requires knowledge of the whole network via routing-path or neighbourhood coordination. For example, Li *et al.* in [3] targeted on minimizing the remaining traffic based on popularity of contents. Their policy relied on statistics collected from lower tier routers along the name resolution path. While the implicit caching policy works independently on each node. For example, Suksomboon *et al.* proposed

978-1-4673-6432-4/15/$31.00 ©2015 IEEE

a probabilistic caching scheme [4] in which the probability of caching is calculated according to request frequency of contents received by each router. In contrast to described work, our proposed strategy addresses the issue of the management of cache under video delivery scenarios.

The problem of video caching has been addressed in previous literature as a core challenge in the Internet under HTTP. Thus, Dynamic Adaptive Streaming over HTTP (DASH) [5] was developed to handle heterogeneous network condition. The DASH rate adaptation algorithm working on clients' devices is designed to change requested video rate on-the-fly based on users' available bandwidth. Therefore, some research (*e.g.*, [6], [7]) has been devoted to that adaptation strategy, considering playback smoothness, average quality, playback interruption, etc. However, these attempts are largely HTTP based, and do not address ICN architectures with intrinsic caching capabilities. Grandl *et al.* in [8] pointed out the issue of caching competition between contents in different bit rates in CCN. Thus they proposed *DASH-INC* which only keeps the highest rate in cache and performs transcoding once a lower rate is requested. However, with the increasing demand in video traffic, one cannot simply assume that in-node processing for transcoding would be scalable for real-time requests. Our work tackles this problem in a radically different way by caching contents with multiple bit rates according to their popularity.

## III. *DASCache* Caching Modeling

It is important to explain the design of caching strategies in light of the operation of ICNs, and their intrinsic in-network caching properties. In this section, we will elaborate on the proposed *DASCache* cache management strategy, and detail the assumptions and axioms upon which our system is built. More importantly, the objective of minimizing access time while improving throughput is explained in terms of the facilitated improvement in users' QoE.

### A. System Description and Axioms

We build *DASCache* over CCN architecture. It is important to note that our proposed strategy is designed to address the primitives of ICN paradigms in general, and is not specific to CCN. However, we choose to address CCN as a viable use case to demonstrate utility and benchmark to current state-of-the-art caching models in the literature. This is further elaborated upon in the Performance analysis, Section V.

Our system primarily targets video delivery in ICNs. All packets in the envisioned CCN scenario are assumed to be related to video contents and the size of network is encompassed by a single ISP [3]. However, our solution could be generalized for multiple (co-existing) ISPs by considering cache partitioning and multi-homing. To maintain a concrete description, we will opt for analyzing and presenting *DASCache* in light of a single ISP.

There are three different types of routers: the edge router, intermediate router and a single gateway. We assume all chunks are kept in a repository (main content producer) which is reachable via the ISP's gateway over a high bandwidth link; mimicking a common scenario in real world deployments and envisioned ICN architectures. All clients are served by edge routers; intermediate routers never connect with clients directly. In this model, node failure is not considered. If there is any router which is not accessible, *DASCache* just needs to refresh the routing topology and runs for a new cache placement solution.

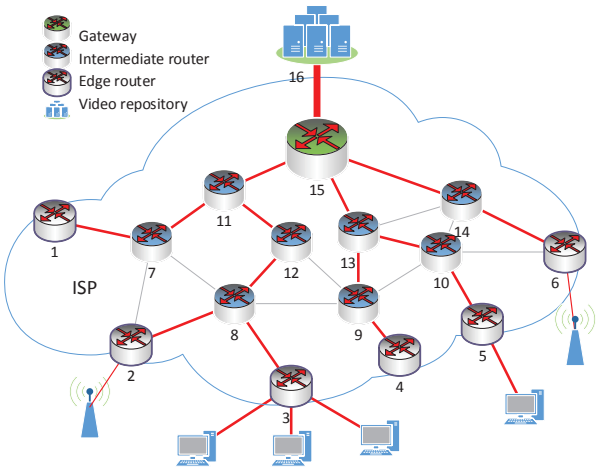



Fig. 1: Network Topology

Although CCN supports multi-path routing which increases the chance of cache hits, the overhead studied by [9] demonstrated performance degradation at the same time. In order to contain our model, we adhere to CCN's default setting: single-path routing without cooperative schemes that get an extra complication in building the model. Our proposed solution is indifferent to network topology. However, we will adopt a tree routing topology as shown in Fig. 1 (marked in red lines) in our modelling to enable a rigorous analysis of network delay.

*DASCache* works in rounds. As to each round, a monitoring window set by network provider is responsible for collecting statistics at each edge router such as users' request frequency and current available bandwidth. Based on this aggregated data, the network provider would predict the bit rate chosen by clients in the next round. *DASCache* is triggered to run for an optimal cache allocation to tune the content placement according to this predicted information at the end of each round. Once the placement is determined, the contents in cache keep unchanged during that round.

This model allows for variations in requested bit rates for every user, since rate adaptation algorithm should cater to individual requests by each user. However, observed that users under the service of the same edge router would share similar network condition, for the reason of simplicity, we assume that users under the service of the same edge router would all choose the same bit rate (which is the most requested one).

### B. Notations and Assumptions

All nodes in the proposed network are modelled as a connected graph $G = (V, E)$ in which node in $V$ are composed of a set of Edge nodes $N$, intermediate nodes $I$ and a single gateway $R$. Node $j$, where $j = 1, \ldots, |V|$ is equipped with content storage with capacity $C_j$ which represents the class-specific capacity dedicated to video caching. How to decide $C_j$ belongs to the *cache space allocation* problem which is beyond the scope of our model.

Video files are divided into $K$ chunks, according to a fixed time interval, and each chunk is identified by a unique number. Chunk $k$ is requested by clients with probability $\{q_k\}_{k=1,\ldots,K}$.

TABLE I: Notation

| | |
|---|---|
| $V$ | CCN routers within the ISP |
| $N$ | Edge routers |
| $I$ | Intermediate routers |
| $R$ | Gateway |
| $C_j$ | Cache capacity (size) of node $j$ |
| $q_k$ | Popularity distribution of requests for chunk $k$ |
| $\pi_j$ | Binary indicator of requested bit rate at edge router $j$ |
| $S$ | Video chunk size |
| $x_j$ | Cache deployment configuration matrix at router $j$ |
| $\lambda_j$ | Interest request arrival rate matrix at router $j$ |
| $\lambda'_j$ | Sum of users' request rate received by edge router $j$ |
| $L_j$ | Set of nodes on routing path from edge router $j$ |
| $VRTT_k(j)$ | Virtual round trip time delay from edge router $j$ |
| $D(i,j)$ | two-way delay between router $i$ and $j$ |
| $\theta_{ij}$ | Downlink bandwidth between router $i$ and $j$ |
| $Q_{ij}$ | Queueing delay between router $i$ and $j$ |
| $\mu_j$ | Interest miss rate matrix at router $j$ |
| $\rho_{ij}$ | Traffic load on link between router $i$ and $j$ |

There are $B$ bit rates available for request while a vector $S_{1\times B}$ is used to denote the size of chunks encoded with different bit rates. For example, $S(b), b = 1,\ldots,B$ denotes an element in vector $S$ representing chunk size with bit rate $b$. Hence, each video chunk now is identified by a two-dimensional index $(k,b)$.

$\{(\pi_{B\times 1})_j\}_{j=1,\ldots,|N|}$ is defined as a binary indicator vector for edge router $j$, in which element $\pi_j(b) \in \{0,1\}, b = 1,\ldots,B$. If $\pi_j(b) = 1$, it indicates that bit rate $b$ is requested by clients at edge router $j$. As we assume that clients served by the same edge router will request for the same bit rate, thus we have $\sum_{b=1}^{B} \pi_j(b) = 1$. Let matrix $\{(x_{K\times B})_j\}_{j=1,\ldots,|V|}$ denote our cache deployment configuration, where $x_j(k,b) \in \{0,1\}, k = 1,\ldots,K; b = 1,\ldots,B$. $x_j(k,b) = 1$ indicates that the video chunks indexed by $(k,b)$ should be cached at node $j$ during the next round.

Users' requests received by each router is assumed to be following Poisson Process. According to the superposition property, the interest arrival rate on intermediate routers is the sum of rates of its children nodes. We denote the interest arrival rate matrix at node $j$ with $\{(\lambda_{K\times B})_j\}_{j=1,\ldots,|V|}$. The element at row $k$ column $b$ represents request rate for video chunk with index $(k,b)$. As for edge router $j$, let $\lambda'_j$ denote the request rate made by all clients and $b_j$ the chosen bit rate. Thus, we have $\lambda_j(k,b_j) = \lambda'_j q_k$.

$\{L_j\}_{j=1,\ldots,|N|}$ is a sequence of node IDs. Elements in this sequence are those which receive interest packets for video chunks sent from edge router $j$. In other words, $\{L_j\}$ contains nodes located on the routing path starting from edge node $j$. We define $L_j(1) = j$ and $L_j(i+1)$ always denotes the ID of the next hop node of $L_j(i)$ on the routing path of *Interest* packet. The last element in $\{L_j\}$ is the video repository accessed by the ISP. For example, following the topology depicted in Figure 1 and considering $j = 2$, $L_2$ will be $(2,8,12,11,15,16)$.

To summarize, Table I lists all notations used in our model. Please note that those symbols that have not been explained earlier but will be presented later are included as well.

### C. Problem Model

In adaptive streaming, throughput is a fundamental metric in rate adaptation algorithms to estimate maximal supported bit rates by measuring the round trip time (RTT) delay of video chunks. We argue for approximating the maximal average throughput a user could achieve by minimizing the average access time per bit. Therefore, the optimization objective could be represented as:

$$\min \quad \sum_{j=1}^{|N|} \frac{\mathbb{E}[\textit{AccessTimePerBit}(j)]}{|N|} \tag{1}$$

To calculate $\mathbb{E}[\textit{AccessTimePerBit}(j)]$, we first define the virtual round trip time (VRTT) delay in Eq 2 where $D(i,j)$ denotes the value of two-way delay among router $i$ and $j$. Then, the VRTT of video chunk $k$ requested by clients under service of edge router $j$ is:

$$\mathbb{E}[VRTT_k(j)] = \sum_{i=1}^{|L_j|-1} \Big(\mathbb{E}[D(L_j(i), L_j(i+1))]\Big(1 - \max_{m=1,\ldots,i} x_{L_j(m)}(k,b)\Big)\Big) \tag{2}$$

To elaborate, we take $j = 2$ as an example from Fig.1. If there is no cache in the network, $\mathbb{E}[VRTT_k(2)]$ is the sum of two-way delay on links between routers $(2,8),(8,12),(12,11),(11,15)$ and $(15,16)$. However, whether data packets actually travel through the link depends on the cache on the routing path. For example, delay between router $(12,11)$ should not be add into $\mathbb{E}[VRTT_k(2)]$ when the video chunk $k$ has already cached in router 2, 8 or 12. If any binary indicator among $x_2(k,b), x_8(k,b), x_{12}(k,b)$ equals 1, $\mathbb{E}[D(L_2(12), L_2(11))]$ will not be added. The summation in Eq 2 is to add up the average delay on each link of routing path from edge router until the node on which the interest is satisfied.

The expected access time per bit for chunk k requested from edge router $j$ is denoted as,

$$\mathbb{E}[\textit{AccessTimePerBit}_k(j)] = \frac{\mathbb{E}[VRTT_k(j)]}{S \cdot \pi_j} \tag{3}$$

Such that,

$$\mathbb{E}[\textit{AccessTimePerBit}(j)] = \sum_{k=1}^{K} q_k \times \frac{\mathbb{E}[VRTT_k(j)]}{S \cdot \pi_j} \tag{4}$$

Hence, the optimization objective in Eq 1 is formally represented as:

$$\min \quad \sum_{j=1}^{|N|} \sum_{k=1}^{K} q_k \times \frac{\mathbb{E}[VRTT_k(j)]}{|N| S(b_j)} \tag{5}$$

where $b_j$ is subject to $\pi_j(b_j) = 1$.

### D. Queueing model and Derivations

In order to compute the $VRTT_j(k)$ in Eq 2, we make a quantitative analysis for each link and calculate the $D(i,j)$. Since processing and propagation delay are negligible, we assign a representative value $P$ to denote them. Let $\theta_{ij}$ denote the downlink bandwidth between node $i$ and $j$ (data packet from $j$ to $i$). When video chunk with bit rate $b'$ is transmitting, the average two-way delay between node $i$ and $j$, $\mathbb{E}[D(i,j)]$, consisting of propagation, processing, transmission and queueing delay, is calculated as

$$\mathbb{E}[D(i,j)] = P + \frac{S(b')}{\theta_{ij}} + \mathbb{E}[Q_{ij}] \tag{6}$$

Data packets, which is delivered according to user requests, arrive at each router following a Poisson Process with memoryless property. Under the video delivery scenario, the size of packets encoded with a given bit rate is equivalent to each other because each packet contains the same length of playback time. Such that, the job service time of a packet in

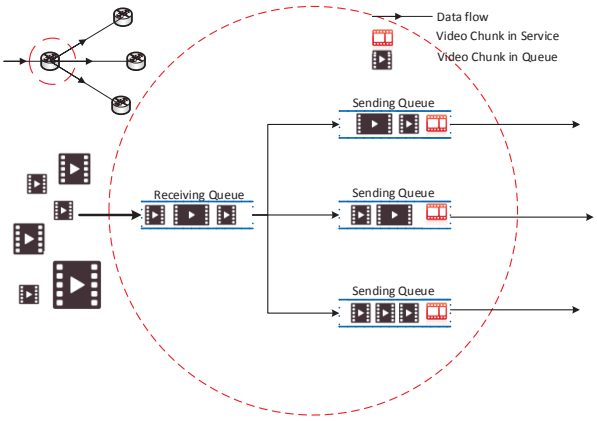



Fig. 2: Queueing model for Adaptive Streaming

the queueing system is always chosen from a set of discrete values determined by available bit rates and the service time for packets encoded with the same bit rate is deterministic. Therefore, we employ a multiclass M/G/1 queueing model to calculate the queueing delay, $\mathbb{E}[Q_{ij}]$.

As video streaming data dominates total network traffic, other traffic types, (*e.g.*, web pages, emails) have both smaller proportion and packet sizes compared to video contents. According to the Proposition III.1, traffic of these types can generate quite limited contribution on average queueing delay. Therefore, this model only considers video packets in the queueing system.

To use this model, we must ensure that there are no overloaded links within the network. Denote with $\rho_{ij}$ the traffic load between node $i$ and $j$ and thus $\rho_{ij} < 1$ must be satisfied. However, under the setting of dynamic adaptive streaming, this condition is guaranteed to be satisfied in the end. The overload caused by increased number of users' requests would be detected by the rate adaptation algorithm since a lot of video packets would be queued or dropped in the network which results in a low or even zero throughput. This algorithm will then control the user's device to request for lower bit rates which will reduce the amount of data delivered within the network and effectively relieve the load.

Denote with router $H_j$ a set of children nodes of router $j$ in tree topology. We have the following proposition to derive $\mathbb{E}[Q_{ij}]$.

**Proposition III.1.** *If $\rho_{ij} < 1$, $\forall j \in V, \forall i \in H_j$, the average queueing delay for data packets delivered from router $j$ to $i$ is*

$$\mathbb{E}[Q_{ij}] = \frac{1}{2}\frac{\sum_{b=1}^{B}\mu_i(\cdot,b)\mathbb{E}[\varphi_{ij}(b)]^2}{1-\sum_{b=1}^{B}\mu_i(\cdot,b)\mathbb{E}[\varphi_{ij}(b)]} \tag{7}$$

*Proof:* The queueing in our video delivery scenario is illustrated in Fig.2. According to the superposition property, the rate of interest packet which request for bit rate $b$ missed by caches in router $i$, $\mu_i(\cdot,b)$ is given by,

$$\mu_i(\cdot,b) = \sum_{k=1}^{K}\lambda_i(k,b)(1-x_i(k,b)) \tag{8}$$

$$\lambda_i(k,b) = \sum_{m=1}^{\|H_i\|}\mu_{H_i(m)}(k,b) \tag{9}$$

Eq 8 and 9 represent the relation between interest arrival and interest miss rate. Take Node 11 in Figure 1 as an example, Eq 9 represents that the interest packet arrival rate on Node 11 is the sum of rates missed by caches in Node 7 and 12. Eq 8 calculates the interest miss rate on Node 11 based on the arrival rate and whether requested contents are cached on Node 11 or not. If it is cached (which means $x_{11}(k,b) = 1$), the corresponding request could be immediately satisfied and there will be no need to forward this interest packet. As node $j$ is responsible for forwarding data packet along the link to the node $i$ according to the 'bread crumb' left by interest packet, $\mu_i(\cdot,b)$ is also used as the data (encoded with bit rate $b$) input rate to the buffer in router $j$. The expected job service time of data packets encoded with bit rate $b$, $\mathbb{E}[\varphi_{ij}(b)]$ is given by,

$$\mathbb{E}[\varphi_{ij}(b)] = \varphi_{ij}(b) = \frac{S(b)}{\theta_{ij}} \tag{10}$$

and then, the traffic load $\rho_{ij}$ is,

$$\rho_{ij} = \sum_{b=1}^{B}\rho_{ij}(b) = \sum_{b=1}^{B}\mu_i(\cdot,b)\mathbb{E}[\varphi_{ij}(b)] \tag{11}$$

when $\rho_j < 1$, it means that the input rate to the queue is less than the output rate. Hence, the model is in stable status. In other words, the requested resource will never exceed the maximum that the network can provide.

To calculate $\mathbb{E}[Q_{ij}]$, we apply Little's Theorem and extend *Pollaczek-Khinchin (P-K) formula* [10]. In P-K formula, the expected waiting time in queue for the *i*th packet is the sum of service time of any previous packets which arrive before *i*th packet and the Residual Service time ($RS_{ij}$). $RS_{ij}$ represents the remaining time seen by the new packet when it arrives until current in-service packet is complete. Let us denote $N_Q(b)$ the number of data packets waiting in the queue which are encoded with bit rate $b$. Such that, the expected queueing delay $\mathbb{E}[Q_{ij}]$ is,

$$\begin{aligned}\mathbb{E}[Q_{ij}] &= \mathbb{E}[RS_{ij}] + \sum_{b=1}^{B}\mathbb{E}[N_Q(b)]\mathbb{E}[\varphi_{ij}(b)] \\ &= \frac{\mathbb{E}[RS_{ij}]}{1-\rho_{ij}}\end{aligned} \tag{12}$$

To apply in multiclass scenario, $\mathbb{E}[RS_{ij}]$ [10] is,

$$\mathbb{E}[RS_{ij}] = \sum_{b=1}^{B}\frac{\mu_i(\cdot,b)\mathbb{E}[\varphi_{ij}(b)^2]}{2} \tag{13}$$

Because service time of data packets, $\varphi_{ij}(b)$ shown in Eq 10, is a deterministic value, we will have $\mathbb{E}[\varphi_{ij}(b)^2] = \mathbb{E}[\varphi_{ij}(b)]^2$. Therefore, synthesis Eq 11 12 13, queueing delay in Eq 7 is proven. ■

## IV. *DASCache* Caching Solution

Our objective in *DASCache* is to find the optimal video content placement, in every round, which minimizes the average access time per bit to all users and thereby increases their respective throughput. A solver is called at the end of every round, as highlighted in Section III, based on aggregated popularity predictors and changes in traffic, to find an optimal cache assignment. In order to formulate a feasible optimization problem, we explain a method in this section which transforms the objective given in Eq 5 to a binary integer programming (BIP) problem.

Synthesizing Eq 2 and 5, we apply the '**big-M**' method ('**M**' denotes a very large positive number) to obtain a

standard form of binary integer programming by substituting $\max_{m=1,\ldots,i} x_{L_j(m)}(k,b)$ in Eq 2 with an artificial binary variable $m_j(i)$ and adding another two constraints in Eq 18. Our optimization problem is finally formulated as follows,

$$\min \quad \sum_{j=1}^{|N|}\sum_{k=1}^{K}\sum_{i=1}^{|L_j|-1}\frac{q_k\mathbb{E}[D(L_j(i),L_j(i+1))](1-m_j(i))}{|N|S(b_j)} \tag{14}$$

$$s.t. \quad \forall j \in [1..|N|], \quad \forall l \in [1..|V|], \quad \forall i \in [1..|L_j|-1] \tag{15}$$

$$b_j = arg(\pi_j(b)=1) \tag{16}$$

$$m_j(i) = \max_{m=1,\ldots,i} x_{L_j(m)}(k,b_j) \tag{17}$$

$$m_j(i) \leq \begin{cases} x_{L_j(i)}(k,b_j) + \mathbf{M}n_j(i) \\ m_j(i-1) + \mathbf{M}\left(1-n_j\left(i\right)\right) \end{cases}, \forall i > 1 \tag{18}$$

$$\sum_{k\in K}\sum_{b\in B} W_l(b)x_l(k,b) \leqslant C_l, \tag{19}$$

$$W_l(b) = \begin{cases} S(b_j) & \textit{if} \quad l \in L_j \\ +\infty & \textit{if not} \end{cases} \tag{20}$$

$$m_j(i), n_j(i), x_l(k,b) \in \{0,1\} \tag{21}$$

Eq 19 constitutes the cache capacity constraint in optimization which ensures the size of cached video objects would not exceed the cache limit. $W_l(b)$ represents required storage space to assign a video chunk encoded with bit rate $b$ to node $l$ in $L_j$. In Eq 20, $W_l(b)$ equals to video chunk size only on those routers which interest requests for bit rate $b$ ever reach.

To ensure $\mathbb{E}[D(L_j(i-1),L_j(i))]$ in Eq 14 is a constant value, the optimization must run iteratively such that the entire problem could be solved as a linear programming. It is because $\mathbb{E}[D(L_j(i-1),L_j(i))]$ is given by Eq 6 and 7 while cache configuration ($x_i(k,b)$) may influence the interest miss rate ($\mu_i(\cdot,b)$) and then change the value of $\mathbb{E}[D(L_j(i-1),L_j(i))]$. As a solution, we apply the result of $x_i(k,b)$ in the previous run to queueing model and achieving a deterministic value of $\mathbb{E}[D(L_j(i-1),L_j(i))]$. Subsequent runs constantly tune the cache configuration and this algorithm will stop if two consecutive runs generate the same/close results.

## V. Performance Analysis

In this section, the performance of our *DASCache* caching strategy is presented and evaluated.

### A. Experiment Setup

We build the simulation environment through customization on ns-3 based simulator, ndnSIM [11]. In order to give optimal caching deployment, we also implement an ILP solver in which calls Gurobi [12] engine to solve Eq 14.

Videos are segmented into chunks with duration of 10 seconds. Within the experiment, we consider five available bit rates and the size for each one is determined according to sample videos [13]. The popularity distribution is assumed to be Zipf-like [14] and the chunk $i$ is requested with the probability $p_i = \frac{\beta}{i^\alpha}$, where $\beta = (\sum_{i=1}^{N} i^\alpha)^{-1}$. The value of $\alpha$ represents the skewness factor. A large $\alpha$ means less video chunks have similar popularity.

TABLE II: DEFAULT PARAMETERS FOR SIMULATION

| Parameters | Value |
|---|---|
| # of video chunks | 3200 |
| Video period(s) | 10 |
| # of routers in ISP | 60 |
| # of edge routers | 35 |
| Skewness factor $\alpha$ | 1.2 |
| Available bit rates | 5 |
| Size of bit rate chunks(KB) | {332, 390, 454, 1364, 2465} |
| Bandwidth between gateway and video repo(Gbps) | 1 |
| Bandwidth between routers within ISP(Mbps) | 20 |
| Simulation period (s)/run | 2000 |

*DASCache* caching management targets optimizing *access time per bit*. However, the ultimate goal is to improve the throughput measured by users. Therefore, we use *Average Throughput* to measure the performance. It is calculated by having video chunk size divided by average data packet retrieval time which is measured between two timestamps when interest packet is sent and corresponding data packet arrives at user's device.

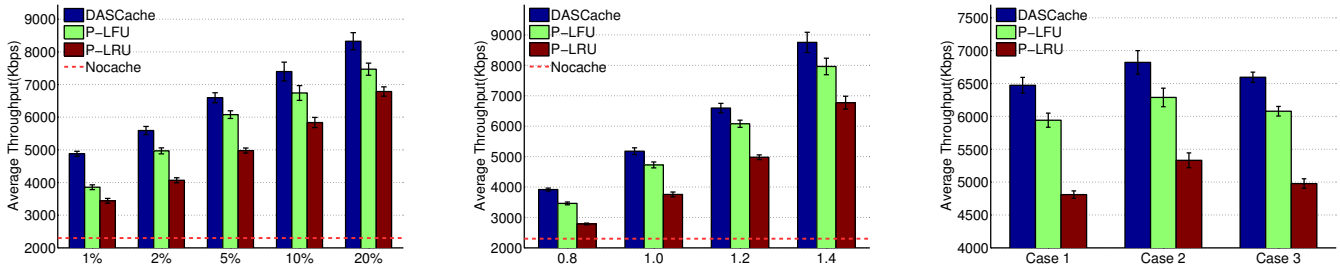
Users' choices of bit rates, guided by adaptation algorithm, must not exceed the upper limit of available bandwidth and are constrained by the network bottleneck. Thus we make the following traffic generation rule: I. Random bit rate is chosen and assigned to each client. We regarded it as the predicted one made by ISP provider. II. Choose initial average request rate ($\lambda$) for clients to ensure that the traffic load on the link is close to full load between edge router and its direct parent node. III. Calculate traffic load ($\rho$) on all links in the routing topology according to Eq 11, starting from those close to edge routers till the gateway. If $\rho < 1$ is not satisfied on some link, we decline users' request rate ($\lambda$) proportionally. Repeat step III. until $\rho < 1$ is satisfied on all links. This rule will make some links become bottlenecks in the network and let the predicted bit rate reasonable in the experiment.

### B. Performance evaluation

As our *DASCache* method relies on collected statistics and only refreshes cache once in a round. In order to make fair comparisons, we use periodic LRU (P-LRU) and periodic LFU (P-LFU). P-LRU is modified according to the Least Recently Used (LRU) scheme but the replacement only occurs at the end of each round based on the request sequence it records. Similar change is made to P-LFU which is from Least Frequently Used (LFU) scheme.

*1) Impact of total cache capacity:* To test the impact of cache capacity, we assume homogeneous cache allocation on routers and define a multiplier $\zeta$ $(0 < \zeta \leq 1)$. The cache capacity of each router, $C_j$, is then defined as, $C_j = \zeta K|V|^{-1}\sum_{i=1}^{B} S(i)$ which is set to be relative to the total size of videos of all bit rates.

Fig. 3(a) shows the performance of caching when $\zeta$ ranges from 1% to 20%. Our *DASCache* deployment outperforms two other caching schemes among all tested cases. For example, when $\zeta = 1\%$, our method has 26%, 42%, 112% improvement on average throughput over P-LFU, P-LRU and Nocache respectively. P-LRU yields the worst performance since it only records the most recent subscriptions and cannot distinguish between popular and unpopular requests. The outcome of P-LFU and *DASCache* are close but performance difference lies in the fact that *DASCache* optimizes the data delivery considering the delay and storage utilization in a synthesized manner. In general, all three caching schemes provide much




(a) Cache Percentage ($\zeta$) (b) Skewness Factor ($\alpha$) (c) Cache Capacity Allocation

Fig. 3: Simulation results

improvement compared to the no cache situation.

*2) Impact of popularity skewness factor:* We explore the impact of popularity skewness factor ($\alpha$) on performance of caching with fixed cache capacity $\zeta = 5\%$ and also choose homogeneous cache allocation.

Fig. 3(b) indicates that our *DASCache* strategy achieves the best average throughput and the smallest RTT delay compared with P-LRU and P-LFU schemes over all tested values of $\alpha$. For example, when $\alpha = 0.8$, our strategy achieves improvement of 13% and 41% on throughput compared with P-LRU and P-LFU respectively. When popularity skewness increases, we observe that all caching schemes achieve significant performance improvement. The reason is that the amount of popular data is smaller as $\alpha$ is larger, thus more of popular video segments can be kept in the cache and respond to requests at a faster speed.

*3) Impact of cache capacity allocation:* Instead of focusing on homogeneous cache allocation, in this experiment, we also test two simple heterogeneous cases. Let us denote that Case 1: intermediate routers have the cache capacity twice than edge routers; Case 2: edge routers have the cache capacity twice than intermediate routers; Case 3: Same cache capacity among all routers (homogeneous case).

Shown in Fig. 3(c), we observe that in Case 2, all caching schemes achieve better performance than two other scenarios. The result is straightforward since more interest requests could be satisfied closer to users when the cache capacity on edge routers is larger. However, it is worthwhile to note that the performance difference is negligible. The *DASCache* deployment in Case 2 is only 348Kbps(5.4%) and 225Kbps(3.4%) better than Case 1 and 3 on average. The underlying reason is that redundant cached contents exhaust space for caching new objects which neutralizes the benefit of moving more contents closer to users. This result also shows insight that cache capacity allocation has less influence compared with content deployment under this particular setting.

## VI. Conclusion

In this paper, we proposed our cache management strategy, *DASCache*, which improves the QoE of users over ICN. The novelty in *DASCache* stems from achieving optimal cache assignment considering dynamic adaptive streaming. In the proposed strategy, simulation results demonstrated how *DASCache* outperforms two mainstream caching schemes.

In future work, instead of requiring global knowledge from network provider, we will develop a distributed algorithm to cater for scenarios with dynamic network conditions (e.g. frequent bit rate switch by user), and to scale with larger networks. As our formulation is based on an LP which is inherently NP-hard, we advocate for extending this work with dynamic heuristics that could operate at quasi-optimality in the large scale.

## References

[1] Cisco, "Cisco visual networking index: Forecast and methodology, 2011–2016," *CISCO White paper*, pp. 2011–2016, 2012.

[2] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*. ACM, 2009, pp. 1–12.

[3] J. Li, H. Wu, B. Liu, J. Lu, Y. Wang, X. Wang, Y. Zhang, and L. Dong, "Popularity-driven coordinated caching in named data networking," in *Proceedings of the eighth ACM/IEEE symposium on Architectures for networking and communications systems*, 2012, pp. 15–26.

[4] K. Suksomboon, S. Tarnoi, Y. Ji, M. Koibuchi, K. Fukuda, S. Abe, N. Motonori, M. Aoki, S. Urushidani, and S. Yamada, "Popcache: Cache more or less based on content popularity for information-centric networking," in *Local Computer Networks (LCN), 2013 IEEE 38th Conference on*, 2013, pp. 236–243.

[5] T. Stockhammer, "Dynamic adaptive streaming over http–: standards and design principles," in *Proceedings of the second annual ACM conference on Multimedia systems*, 2011, pp. 133–144.

[6] S. Xiang, L. Cai, and J. Pan, "Adaptive scalable video streaming in wireless networks," in *Proceedings of the 3rd multimedia systems conference*. ACM, 2012, pp. 167–172.

[7] B. Wang, J. Kurose, P. Shenoy, and D. Towsley, "Multimedia streaming via tcp: An analytic performance study," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, vol. 4, no. 2, p. 16, 2008.

[8] R. Grandl, K. Su, and C. Westphal, "On the interaction of adaptive video streaming with content-centric networking," *arXiv preprint arXiv:1307.0794*, 2013.

[9] G. Rossini and D. Rossi, "Evaluating ccn multi-path interest forwarding strategies," *Computer Communications*, vol. 36, no. 7, pp. 771–778, 2013.

[10] D. P. Bertsekas, R. G. Gallager, and P. Humblet, *Data networks*. Prentice-hall Englewood Cliffs, NJ, 1987, vol. 2.

[11] A. Afanasyev, I. Moiseenko, L. Zhang *et al.*, "ndnsim: Ndn simulator for ns-3," *University of California, Los Angeles, Tech. Rep*, 2012.

[12] Gurobi, "Gurobi optimizer reference manual," http://www.gurobi.com/documentation/5.6/reference-manual/, accessed on 2014-10-15.

[13] Youtube, "Mpeg-dash/media source demo," http://dash-mse-test.appspot.com/media.html, accessed on 2014-10-15.

[14] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and zipf-like distributions: Evidence and implications," in *INFOCOM*, vol. 1. IEEE, 1999, pp. 126–134.