Abstract—The exploding volumes of mobile video traffic call for deploying content caches inside mobile operators network. With in-network caching, users’ requests for popular content can be served from a content cache deployed at mobile gateways in vicinity to the end user, therefore considerably reducing the load on the content servers and the backbone of operator’s network. In practice, content caches can be installed at multiple levels inside an operator’s network (e.g., serving gateway, packet data network gateway, RAN, etc.), leading to an idea of hierarchical in-network video caching. In order to evaluate the pros and cons of hierarchical caching, in this paper we formulate a cache provisioning problem which aims to find the best trade-off between the cost of cache storage and bandwidth savings from hierarchical caching. More specifically, we aim to find the optimal size of video caches at different layers of a hierarchical in-network caching architecture which minimizes the ratio of transmission bandwidth cost to storage cost. We overcome the complexity of our problem which is formulated as a binary-integer programming (BIP) by using canonical duality theory (CDT). Numerical results obtained using the invasive weed optimization (IWO) show that important gains can be achieved, with benefit-cost ratio and cost efficiency improvements of more than 43% and 38%, respectively.

Index Terms—Cache storage, canonical duality, hierarchical in-network caching, invasive weed optimization, mobile video delivery.

I. INTRODUCTION

The extensive growth in adoption of smartphones and tablets has led to a continuous increase in mobile video traffic. According to the recent reports [1], mobile video will represent 72% of global mobile data traffic by 2019, a 13-fold increase from 2014. This new phenomenon has urged mobile operators to redesign their networks and search for cost-effective solutions to bring content closer to the end user [2], [3].

One approach to this problem lies in installing geographically distributed content delivery networks (CDNs), which can efficiently serve users within certain geographic areas. However, in order to reach an end-user’s device, CDN-served traffic must still traverse through the wireless carrier core network (CN) and radio access network (RAN) both of which contribute to delays in streaming video content. In contrast, with in-network caching, users can access popular content from caches of nearby mobile network operator (MNO) gateways [i.e. evolved packet core (EPC) and RAN] [3]–[8], therefore significantly reducing video streaming latency. Moreover, from the Internet Service Providers’ (ISP) perspective, in-network caching also helps to reduce inter- and intra-ISP traffic and, so, to optimize operating costs for leasing expensive fiber lines that connect eNodeBs to EPC [7], [8].

Several approaches have been proposed to analyze intelligent caching strategies for mobile content caching inside MNO’s network [5]–[7]. An extensive overview of the techniques for in-network content caching in 5G mobile networks has been introduced in [8], whereas different proactive mobile caching schemes have been discussed in [4], [9]. The current paper contributes to this stream of work by analyzing the trade-off between the potential savings from- and infrastructural costs of hierarchical in-network caching. In more details, the main contributions of this paper can be summarized as follows:

- To the best of our knowledge, we present the first attempt to formalize the problem of storage provisioning for a hierarchical in-network video caching which optimizes the trade-off between the cost of transmission bandwidth and the cost of storage.
- We focus our analysis on Scalable Video Coding (SVC)-based dynamic adaptive streaming over HTTP (DASH) format which encodes a video into different quality layers and is therefore more resource-efficient than traditional H.264/AVC-based DASH in which a separate AVC video file is encoded for each video quality format [10].
- We solve the storage provisioning problem using CDT [11]. More specifically, we transform our BIP problem into a canonical dual problem in continuous space, which is a concave maximization problem. Additionally, we provide the conditions under which the solutions of the canonical dual problem and primal problem are identical.
- The canonical dual problem results in complex non-linear equations which are efficiently solved by applying IWO algorithm [12].

In summary, our results suggest an improvement of up to 43.74% in benefit-cost ratio and 38.59% in cost efficiency in comparison with a naive Least Frequently Used (LFU) approach.

The rest of the paper is structured as follows. Section II describes the system model. The cache provisioning problem is formulated in III. Section IV presents the canonical dual framework. Section V conducts a simulation analysis of the model. The conclusion is presented in Section VI.

II. SYSTEM MODEL

The system consists of I video streams, which are indexed by the set \( I \triangleq \{1, \ldots, I\} \). We index different quality layers of a video stream by the set \( J \triangleq \{1, \ldots, J\} \). By \( q_{ij} \), we denote the \( j \)th quality layer of video \( i \), which
III. Problem Formulation

We formulate the cache provisioning problem as follows.

\[
\max \times \frac{\sum_{n=1}^{N} b_n(l_{nij})}{\sum_{n=1}^{N} c_n(s_n)} \quad (3)
\]

subject to:

\[
s_n = \sum_{i=1}^{I} \sum_{j=1}^{J} f_{ij} x_{nij} \leq m_n \quad \forall n \in N \quad (3a)
\]

\[
\sum_{n=1}^{N} x_{nij} \leq 1 \quad \forall i \in J, \forall j \in J \quad (3b)
\]

\[
x_{nij-1} \geq x_{nij} \quad \forall n \in N, \forall i \in J, \forall j \in J-\{1\} \quad (3c)
\]

\[
x_{nij} \in \{0, 1\} \quad \forall i \in J, \forall j \in J, \forall n \in N. \quad (3d)
\]

The objective of optimization problem (3) is to find the optimal provisioned storage capacity, \(s_n\), which maximizes the ratio of overall benefit (1) to overall cost (2). Constraint (3a) ensures that the cache storage allocated to the \(n^{th}\) layer of the hierarchical caching system is upper-bounded by the maximum possible storage capacity threshold, \(m_n\). Constraint (3b) indicates that each video can be cached in one hierarchical layer inside the in-network caching architecture exclusively. Constraint (3c) ensures that if a video quality layer is cached, all the lower quality layers are cached too. We use binary variables \(x_{nij} \in \{0, 1\}\) explained in section II-A1.

The optimization problem (3) is difficult to solve due to its combinatorial nature. As an intermediate step towards solution, we convert (3) into a BIP problem by defining a cache allocation matrix where instead of making decisions on the basis of individual video quality layer, decisions are made on the basis of feasible set of video layer cache allocation patterns that satisfies constraint (3c). The idea of pattern allocation is similar to [13]. We index all the combinations of video streams and the respective quality layers by the set \(\mathcal{K} \triangleq \{1, \ldots, k, \ldots, K\}\), where each row corresponds to the video stream-video quality layer combination index and each column corresponds to a feasible cache allocation pattern [meeting constraint (3c)]. We denote by \(A\) the total number of feasible allocation patterns. The basic idea of this cache allocation matrix, for the case of 2 videos each with 2 quality layers is illustrated by (4). In any allocation pattern (i.e., any column), a “1” is placed when the video quality layer is cached, otherwise a “0” is placed.

\[
Y^n = \begin{bmatrix}
1 & 1 & 0 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 0 & 1
\end{bmatrix} \quad (4)
\]

We define a cache indicator vector \(x \triangleq [x_n]_{N \times 1}\), where \(x_n = [x_{nai}]_{A \times 1}\). Each entry \(x_{na} \in \{0, 1\}\) indicates whether the cache allocation pattern \(a\) is allocated to hierarchical caching layer \(n\) or not.
Note that all the caches in the hierarchical in-network caching system have the same allocation patterns matrix. We rewrite (3) as a BIP problem as follows:

$$\min_x \left\{ \mathcal{P}(x) = \frac{\sum_{n=1}^{N} \sum_{a=1}^{A} b_{na} x_{na}}{\sum_{n=1}^{N} \sum_{a=1}^{A} c_{na} x_{na}} \right\} \quad (5)$$

subject to:

$$\sum_{a=1}^{A} s_{na} x_{na} \leq m_n \quad \forall n \in \mathbb{N} \quad (5a)$$

$$\sum_{n=1}^{N} \sum_{a=1}^{A} Y_{ka}^n x_{na} \leq 1 \quad \forall k \quad (5b)$$

$$x_{na} (x_{na} - 1) = 0 \quad \forall n \in \mathbb{N}, \forall a \quad (5c)$$

$$\sum_{a=1}^{A} x_{na} = 1 \quad \forall n \in \mathbb{N} \quad (5d)$$

where $b_{na}$ and $c_{na}$ are the transmission bandwidth benefit and storage cost of allocating pattern $a$ to hierarchical cache layer $n$, which results in a provisioned storage size of $s_{na}$. For a cache layer $n$, constraint (5a) puts an upper-bound of $m_n$ on the provisioned storage size, which is equivalent to constraint (3a). Constraint (5b) ensures the exclusivity of the allocated videos, where $Y_{ka}^n$ denotes the $k^{th}$ row and $a^{th}$ column of the matrix $Y^*$, where $Y_{ka}^n = 1$ indicates that the video stream-video quality layer combination $k$ should be cached in hierarchical layer $n$ and $Y_{ka} = 0$ otherwise. Constraint (5c) is a pure binary constraint that ensures $x_{na} \in \{0, 1\}$. Constraint (5d) ensures that at most one allocation pattern is chosen for each caching layer.

Although the optimization problem (5) is simpler and more tractable than (3), the solution is still exponentially complex.

**IV. Canonical Dual Framework**

**A. Dual Problem Formulation**

We convert our BIP problem (5) into a continuous space canonical dual problem using CDT [11], [14], which is solved in continuous space. We then identify the conditions under which the solution of the canonical dual problem is identical to that of the primal. A generic framework for solving 0-1 quadratic problems using CDT can be found in [15]. However, due to additional constraints, our problem is more complex. A framework for solving resource allocation BIP problems using CDT is given in [16], which will be extended to solve (5).

We define the feasible space for the primal problem (5) by $\mathcal{X}_p = \{ x \in \{0, 1\}^{NA} \}$. We temporarily relax the equality constraints (5c) and (5d) to inequalities and transform the primal problem with these inequality constraints into continuous domain canonical dual problem. We then solve the problem in continuous space and provide the conditions under which the solutions of the canonical dual problem and primal problem are identical.

As a key step towards canonical dual formulation, we define the geometrical operator for the primal problem as $\land(y) = (\delta, \beta, \tau, \sigma) \in \mathcal{Y}_g$, which is a vector valued mapping where $\mathcal{Y}_g$ is the feasible space for $y$, and

$$\begin{align*}
\lambda &= \sum_{a=1}^{A} s_{na} x_{na} - m_n N x_1 \\
\mu &= \sum_{n=1}^{N} \sum_{a=1}^{A} Y_{ka}^n x_{na} - 1 |K x_1 \\
\nu &= [x_{na} (x_{na} - 1)] N A x_1 \\
\tau &= [\sum_{a=1}^{A} x_{na} - 1] N x_1
\end{align*} \quad (6)$$

Therefore, the feasible space for $y$ is defined by $\mathcal{Y}_g = \mathbb{R}^N \times \mathbb{R}^K \times \mathbb{R}^{NA} \times \mathbb{R}^N | \lambda \leq 0, \mu \leq 0, \nu \leq 0, \tau \geq 0$.

Next, we define the indicator function [15] as

$$V(y) = \begin{cases} 0 & \text{if } y \leq 0 \\ +\infty & \text{otherwise} \end{cases} \quad (7)$$

We rewrite the primal problem (5) in the canonical form using indicator function (7) as follows:

$$\min \{ V(\land(y)) + \mathcal{P}(x) \} \quad (8)$$

We now define $y^* = (\lambda^*, \mu^*, \nu^*, \tau^*)$ as the vector of dual variables associated with the corresponding restrictions $y \leq 0$. The feasible space for $y^*$ is defined by $\mathcal{Y}_d = \mathbb{R}^N \times \mathbb{R}^K \times \mathbb{R}^{NA} \times \mathbb{R}^N | \lambda^* \geq 0, \mu^* \geq 0, \nu^* \geq 0, \tau^* \geq 0$. Based on the Fechnel transformation, the canonical sup-conjugate function of $V(y)$ is defined as

$$V^*(y^*) = \sup \{ \langle y, y^* \rangle - V(y) | y \in \mathcal{Y}_g, y^* \in \mathcal{Y}_d \}$$

$$= \sup \left\{ \langle \lambda^T \lambda^* + \mu^T \mu^* + \nu^T \nu^* + \tau^T \tau^* - y^* \rangle \right\}$$

$$= \begin{cases} 0 & \text{if } \lambda^*, \mu^*, \nu^*, \tau^* \geq 0 \\ +\infty & \text{otherwise} \end{cases} \quad (9)$$

Using the definition of sub-differential, it can be easily verified that if $y^* > 0$, then the condition $y^T y^* = 0$ leads to $y = 0$, and consequently $x \in \mathcal{X}_p$. Hence, the dual feasible space for the primal problem in (5) is an open positive cone defined by $\mathcal{X}_p^* = \{ y^* \in \mathcal{Y}_d | y^* > 0 \}$.

We define the total complementarity function [11] as

$$\Xi(x, y^*) = \land(x)^T y^* - V^*(y^*) + \mathcal{P}(x), \quad (10)$$

which is obtained by replacing $V(y) = \land(x)^T y^* - V^*(y^*)$ (Fechnel-Young equality) in (8). We use the definitions of $\land(x), V^*(y^*)$ and $\mathcal{P}(x)$ to express $\Xi(x, y^*) = \Xi(x, \lambda^*, \mu^*, \nu^*, \tau^*)$ as given by

$$\Xi(x, y^*) = \sum_{n=1}^{N} \sum_{a=1}^{A} x_{na} \Phi - \sum_{n=1}^{N} \sum_{a=1}^{A} b_{na} x_{na} + \sum_{n=1}^{N} \sum_{a=1}^{A} c_{na} x_{na}$$

$$- \sum_{n=1}^{N} \lambda_n^* m_n - \sum_{k=1}^{K} \mu_k^* - \sum_{n=1}^{N} \tau_n^* + \sum_{a=1}^{A} \nu_a^* x_{na}^2 \quad (11)$$

where $\Phi = \sum_{k=1}^{K} \mu_k^* Y_{ka}^n + \lambda_n^* s_{na} + \tau_n^* - \nu_a^2$. Next, we define the canonical dual function [11], [15] using the canonical dual variables as

$$\Upsilon(\lambda^*, \mu^*, \nu^*, \tau^*) = \text{sta} \{ \Xi(x, \lambda^*, \mu^*, \nu^*, \tau^*) \}, \quad (12)$$
where \( \text{sta}(\cdot) \) denotes finding the stationary point of the function. We are primarily interested in the cache allocation vector \( x \) for a node \( n \). The stationary point of \( \Xi(x, y^*) \) occurs at

\[
x_{na}(y^*) = \frac{1}{2} - \frac{1}{2
u_{na}} \left( \sum_{k=1}^{K} \mu_k Y_{kn} + \lambda^*_n x_{na} + \tau^*_n \right)
\]

\forall n, a,

(13)

where the stationary point is obtained through \( \nabla_x \Xi(x, y^*) = 0 \). Using (12) and (13), we obtain the dual function, which is given by (14), shown at the next page.

The dual function is a concave function on \( \mathbb{R}^p_+ \). The canonical dual problem associated with (5) can be formulated as

\[
\min \left\{ \mathcal{P}(x)|x_p = 0 \right\} = \mathcal{P}(x) = \mathcal{Y}(\lambda^*, \mu^*, \nu^*, \tau^*)\mathbb{X}_{p}^p.
\]

(15)

**Theorem 1.** If \( \mathcal{P}(\hat{x}) = \mathcal{Y}(\hat{y}^*) \) where \( \hat{x} \) denotes the KKT point of the primal problem and \( \hat{y}^* = (\hat{\lambda}^*, \hat{\mu}^*, \hat{\nu}^*, \hat{\tau}^*) \in \mathbb{X}_{p}^p \) denotes the KKT point of the dual function, there exists a perfect duality relationship between the primal problem in (5) and its canonical dual problem.

**Proof.** The proof directly extends from [14]. ■

Theorem 1 shows that the BIP in (5) is converted into a continuous space canonical dual problem which is perfectly dual to it. Moreover, the KKT point of the dual problem provides the KKT point of the primal problem.

**Theorem 2.** (global optimality conditions): If \( \hat{y}^* = (\hat{\lambda}^*, \hat{\mu}^*, \hat{\nu}^*, \hat{\tau}^*) \in \mathbb{X}_{p}^p \), then \( \hat{x} \) is a global minimizer of \( \mathcal{P}(x) \) over \( \mathbb{X}_{p}^p \) and \( \hat{y}^* \) is a global maximizer of \( \mathcal{Y}(\hat{\lambda}^*, \hat{\mu}^*, \hat{\nu}^*, \hat{\tau}^*) \) over \( \mathbb{X}_{p}^p \). Hence, \( \mathcal{P}(\hat{x}) = \min \left\{ \mathcal{P}(x)|x_p = 0 \right\} = \mathcal{Y}(\hat{\lambda}^*, \hat{\mu}^*, \hat{\nu}^*, \hat{\tau}^*) \).

**Proof.** The proof directly extends from [14]. ■

According to Theorem 2, if the given global optimality conditions are met, the solution of the canonical dual problem provides an optimal solution to the primal problem. Solving the KKT conditions associated with the dual function in (14) is necessary and sufficient for global optimality as the dual problem is a concave maximization problem over \( \mathbb{X}_{p}^p \).

The KKT conditions of the dual function in (14) are given by \( \partial \mathcal{Y}/\partial \lambda_{na}^* = 0, \partial \mathcal{Y}/\partial \mu_k = 0, \partial \mathcal{Y}/\partial \nu_{na}^* = 0 \) and \( \partial \mathcal{Y}/\partial \tau_{na}^* = 0 \), where the respective partial derivatives are given by (16)-(19), shown at the next page.

### B. Invasive Weed Optimization Algorithm

Traditional gradient-based algorithms exist in literature for solving the non-linear equations resulting from the KKT conditions associated with the dual function. However, they show many defects such as oscillatory behavior, sensitivity to choice of initial values and complexity associated with the differentiation of KKT conditions and calculation of step size.

We deploy an IWO [12] algorithm for solving the complex non-linear equations associated with the KKT conditions [17]. Inspired by the invasive and robust nature of weeds, IWO is an evolutionary optimization algorithm, which has been shown to perform better than traditional approaches in terms of convergence. It also has the desirable properties of dealing with non-differentiable and complex objective functions and does not show the aforementioned defects.

In summary, the key steps of IWO are **Initialization**, where seeds are randomly dispersed over the search space; **Reproduction**, where every seed grows to a flowering plant and produces seeds; **Spatial Dispersion**, where produced seeds are distributed based on a normal distribution with a mean of zero and standard deviation reducing from an initial value \( \sigma_{\text{initial}} \) to a final value \( \sigma_{\text{final}} \) according to equation \( \sigma_{\text{iter}} = [(\text{iter}_{\text{max}} - \text{iter})/\text{iter}_{\text{max}}]\sigma_{\text{initial}} + \sigma_{\text{final}} \), where \( g \) is the modulation index; and **Competitive Exclusion**, where a competitive mechanism is implemented for eliminating undesirable plants. A detailed discussion on IWO is out of scope of this paper. Interested reader is referred to [12], [18].

**Algorithm 1:** Hierarchical caching based on IWO

```plaintext
initialize \( \lambda^*, \mu^*, \nu^*, \tau^*, \forall n \in \mathbb{N}, \text{iter} = 0; \)
\forall \partial \mathcal{Y}/\partial \nu_{\text{na}^*}, \text{where} \nu_{\text{na}^*} \in (\delta^*, \mu^*, \nu^*, \tau^*)
create initial population of \( Q \) individuals (weeds): \( \mathcal{W} = \{W_1, \cdots, W_Q\} \);
while \( |\nu_{\text{na}^*}| > \varphi \) or \( \text{iter} = \text{iter}_{\text{max}} \) do
evaluate the fitness of each individual i.e., calculate \( f(W_n), \forall n \in \mathcal{W} \);
\text{sort} \mathcal{W} \text{ in ascending order according to} \ f(W_n); 
\text{select the first} \ Q_p \ \text{individuals of} \ \mathcal{W} \text{ to create the set} \ \mathcal{W}_p; 
\forall W_j, j = 1, \cdots, Q_p 
generate \ S_j = \{f(W_j) - f_{\text{max}} \sum (S_{\text{max}} - S_{\text{min}}) + S_{\text{max}} \text{ seeds;} 
\text{create population of generated seeds,} \ \mathcal{W}_s = \{W_s\}; 
for \ i = 1: |\mathcal{W}_s| \ do
\text{\quad} W_{s_i} \leftarrow W_{s_i} + \varphi', \text{where} \ \varphi' \sim L(0, \sigma_{\text{iter}}); 
end
create \ \mathcal{W}^* = \mathcal{W} \cup \mathcal{W}_s; 
\text{sort} \ \mathcal{W}^* \text{ in ascending order according to fitness; select the first} \ Q_{\text{max}} \ \text{individuals of} \ \mathcal{W}^* \text{ and create} \ \mathcal{W};
end
select the best fitted individuals \( \lambda^*, \mu^*, \nu^* \) and \( \tau^* \); calculate \( x_{na} \) using (13);
```

### V. Simulation Results

We assume a hierarchical in-network caching system consisting of 4 layers. Without loss of generality, we initially set the maximum possible storage capacities of hierarchical caching layers 1, 2, 3 and 4 to 200, 400, 500 and 600 gigabytes, respectively. In order to analyze the effects of maximum possible storage capacity on the performance of our proposed approach, we extend the cache size in increments of 20% until the maximum storage capacity of the first, second, third and forth layer caches reach 600, 1200, 1500 and 1800 gigabytes (typical storage capacities available today).

In defining the cost and benefit functions, we assume that the stationary point is obtained through

\[
x_{na}(y^*) = \frac{1}{2} - \frac{1}{2\nu_{na}} \left( \sum_{k=1}^{K} \mu_k Y_{kn} + \lambda^*_n x_{na} + \tau^*_n \right)
\]

\forall n, a.  

(13)
the maximum storage capacity does not lead to a noticeable
videos worthy of being cached. Therefore, further increasing
possible capacity of approximately 3.7 terabytes is reached,
provisioned cache size. We note that when a maximum
increase in the storage capacity increases the identified
storage size under maximum possible capacity varying from
provides a summary of the simulation parameters for IWO.
provisioning algorithm is given as Algorithm 1. TABLE I
the video popularity is Zipf-like with a parameter of 0.6 and
4000 with 3 popular quality layers. As in [19], [20], we assume
benefit. We consider the total number of popular videos to be
allocation vector associated with the dual problem deploying IWO and compute the
Fig. 2 compares the effect of using a logarithmic function
\begin{equation}
\phi = \sum_{n=1}^{N} \sum_{a=1}^{A} \frac{b_{n,a}}{2^{\nu_{n,a}}} + \sum_{n=1}^{N} \sum_{a=1}^{A} \frac{c_{n,a}}{2^{\nu_{n,a}}} - N \sum_{k=1}^{K} m_{n} - N \sum_{n=1}^{N} \tau_{n}.
\end{equation}
\begin{equation}
\frac{\partial \phi}{\partial \lambda_{n}} = - \sum_{n=1}^{N} \sum_{a=1}^{A} \frac{b_{n,a}}{2^{\nu_{n,a}}}, \quad \frac{\partial \phi}{\partial \nu_{n,a}} = \sum_{n=1}^{N} \sum_{a=1}^{A} \frac{c_{n,a}}{2^{\nu_{n,a}}} - N \sum_{n=1}^{N} \sum_{a=1}^{A} \frac{b_{n,a}}{2^{\nu_{n,a}}}, \quad \frac{\partial \phi}{\partial \tau_{n}} = - N.
\end{equation}
more costly and results in more transmission bandwidth saving
benefit. We consider the total number of popular videos to be
4000 with 3 popular quality layers. As in [19], [20], we assume
the video popularity is Zipf-like with a parameter of 0.6 and
the video file sizes follow a Pareto (0.25) distribution with a
minimum size of 60 megabytes.

We solve the KKT conditions for each dual variable associated
with the dual problem deploying IWO and compute the
allocation vector using (13). A pseudo code for the cache
provisioning algorithm is given as Algorithm 1. TABLE I
provides a summary of the simulation parameters for IWO.

Table 1: IWO Numerical Parameter Values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size of initial population ((Q))</td>
<td>20</td>
</tr>
<tr>
<td>Min. fitness threshold ((\phi))</td>
<td>10^{-7}</td>
</tr>
<tr>
<td>Max. no. of iterations ((</td>
<td>iter_{max}</td>
</tr>
<tr>
<td>Max. no. of plants ((Q_{max}))</td>
<td>10</td>
</tr>
<tr>
<td>No. of seeds ((S_{max}, S_{min}))</td>
<td>(5,0)</td>
</tr>
<tr>
<td>Non-linear modulation index</td>
<td>2.5</td>
</tr>
<tr>
<td>Standard deviation ((\sigma_{initial}, \sigma_{final}))</td>
<td>(10,0.01)</td>
</tr>
</tbody>
</table>

Fig. 2 compares the effect of using a logarithmic function
with a linear function in identifying the optimal provisioned
storage size under maximum possible capacity varying from
1.7 to 5.1 terabytes (20% increments). In both scenarios,
an increase in the storage capacity increases the identified
 provisioned cache size. We note that when a maximum
possible capacity of approximately 3.7 terabytes is reached,
the in-network caching system possesses most of the popular
videos worthy of being cached. Therefore, further increasing
the maximum storage capacity does not lead to a noticeable
increase in the provisioned cache size at this point.

We compare our proposed approach with the case when
no storage provisioning is performed within the hierarchical
in-network caching system and popular contents are cached
using least frequently used (LFU) caching algorithm [21]. LFU
 caches the most popular videos in the lower layer caches closer
to the end users [22]. In contrast with the other widely used
caching algorithm, least recently used (LRU), LFU focuses on
historical popularity over a long period of time. As a cache
provisioning technique, our approach also considers a long
term content popularity. Therefore, it is pertinent to compare
Fig. 2: Provisioned storage vs. maximum possible storage.

Fig. 3: Benefit-cost ratio vs. maximum possible storage.

Fig. 4: Storage cost vs. maximum possible storage.

Fig. 5: Provisioned storage of different layers of hierarchical caching system vs. maximum possible storage.

Our proposed scheme with LFU.

Fig. 3 compares the performance of our proposed approach with LFU in terms of benefit to cost ratio under different maximum possible storage capacities mentioned earlier. We note that our proposed approach improves benefit to cost ratio by 43.74%. When there is approximately 3.1 terabytes of storage capacity available, the benefit to cost ratio performance of LFU starts degrading as by this point, most of the popular videos have been cached and adding more storage only increases the cost for the same amount of saving in transmission bandwidth.

Fig. 4 compares the storage cost-effectiveness of our proposed approach with LFU. Since LFU does not support intelligent storage provisioning and uses the maximum storage capacity available, extending the cache size in increments of 20% increases the storage cost exponentially. However, our scheme only uses an optimal portion of the maximum possible storage and hence, decreases the cost significantly. Our proposed approach improves cost-effectiveness by 38.59%. When there is 3.7 terabytes of storage available, the cost starts decreasing in our approach as there is more storage available on cheaper caches at higher layers. Therefore, to increase cost-effectiveness, some of the videos that were previously cached at the expensive lower layer caches move to the higher layers.

Fig. 5 indicates how the increase in the maximum storage capacity affects the provisioned storage size of the caches at each hierarchical layer of the in-network caching system. As more storage is available on the cheaper devices in higher layers, more provisioned storage size is allocated to the higher layer devices due to greater cost efficiency.
Lastly, we discuss the complexity of the IWO algorithm. IWO is an iterative algorithm and is used for each dual variable associated with the dual function in (14). In each iteration for $\chi^* \geq 0, \mu^* \geq 0, \nu^* \geq 0, \tau^* \geq 0$, we compute $N, K, NA, N$, and $\mu$ variables, respectively. Therefore, it has an overall worst case complexity of $O(\text{iter} \cdot (2N + K + NA))$. A detailed performance evaluation of IWO algorithm in terms of convergence and computational time in compare with various algorithms such as Genetic Algorithm and Particle Swarm Optimization can be found in [12] and [18].

VI. CONCLUSION

In this paper, we have proposed a cache provisioning scheme which optimizes cache storage allocation inside a hierarchical in-network caching system in order to minimize both storage and transmission bandwidth costs. We use CDT to convert our BIP problem into its canonical dual. We use the IWO algorithm to obtain the solution of the dual problem. Numerical and simulation results have shown that the proposed scheme outperforms LFU algorithm by more than 43% and 38% in terms of benefit-cost ratio and cost-efficiency improvement, respectively.

ACKNOWLEDGMENT

This work is partially funded by the European Commission Framework Programme 7 Marie Curie Initial Training Networks (ITN) CROSSFIRE project (FP7-PEOPLE-317126).

REFERENCES