Cost-Minimizing Dynamic Migration of Content Distribution Services into Hybrid Clouds

Xuanjia Qiu*, Hongxing Li*, Chuan Wu*, Zongpeng Li† and Francis C.M. Lau*

*Department of Computer Science, The University of Hong Kong, Hong Kong, {xjqiu,hxli,cwu,fcmlau}@cs.hku.hk
†Department of Computer Science, University of Calgary, Canada, zongpeng@ucalgary.ca

***Abstract*—The recent advent of cloud computing technologies has enabled agile and scalable resource access for a variety of applications. Content distribution services are a major category of popular Internet applications. A growing number of content providers are contemplating a switch to cloud-based services, for better scalability and lower cost. Two key tasks are involved for such a move: to migrate their contents to cloud storage, and to distribute their web service load to cloud-based web services. The main challenge is to make the best use of the cloud as well as their existing on-premise server infrastructure, to serve volatile content requests with service response time guarantee at all times, while incurring the minimum operational cost. Employing Lyapunov optimization techniques, we present an optimization framework for dynamic, cost-minimizing migration of content distribution services into a hybrid cloud infrastructure that spans geographically distributed data centers. A dynamic control algorithm is designed, which optimally places contents and dispatches requests in different data centers to minimize overall operational cost over time, subject to service response time constraints. Rigorous analysis shows that the algorithm nicely bounds the response times within the preset QoS target in cases of arbitrary request arrival patterns, and guarantees that the overall cost is within a small constant gap from the optimum achieved by a T-slot lookahead mechanism with known information into the future.**

I. Introduction

Cloud computing technologies have enabled rapid provisioning server utilities to users anywhere, anytime. To exploit the diversity of electricity costs and to provide service proximity to users in different geographic regions, a cloud service often spans multiple data centers over the globe, *e.g.*, Amazon CloudFront, Microsoft Azure. The elastic and on-demand nature of resource provisioning has made cloud computing attractive to providers of various applications. More and more new applications are being created on the cloud platform [1][2], while many existing applications are also considering the cloud-ward move [3][4], including content distribution applications [5][6].

As an important category of popular Internet services, content distribution applications, *e.g.*, video streaming, web hosting and file sharing, feature large volumes of content and demands that are highly dynamic in the temporal domain. A cloud platform with multiple, distributed data centers is ideal to host such a service, with substantial advantages over a traditional private or public content distribution network (CDN) based solution, in terms of more agility and significant cost reduction.

Two major components exist in a typical content distribution application, namely back-end storage for keeping the contents, and front-end web service to serve the requests. Both can be migrated to the cloud: contents can be stored in storage servers in the cloud, and requests can be distributed to cloud-based web services. Therefore, the key challenge for cloud-ward move of a content distribution application is how to efficiently replicate contents and dispatch requests across multiple cloud data centers and the provider's existing on-premise servers such that good service response time is guaranteed and only modest operational expenditure is incurred. Some existing work [3][4][5][6] have advocated to optimize application migration into clouds, but none focuses on guaranteeing over-time cost minimization with a dynamic algorithm.

In this paper, we present a generic optimization framework for dynamic, cost-minimizing migration of content distribution services into a hybrid cloud (*i.e.*, private and public clouds combined), and design a joint content placement and load distribution algorithm that minimizes overall operational cost over time, subject to service response time constraints. Our design is rooted in Lyapunov optimization theory [7][8], where cost minimization and response time guarantee are achieved simultaneously by efficient scheduling of content migration and request dispatching among data centers. Lyapunov optimization provides a framework for designing algorithms with performance arbitrarily close to the optimal performance over a long run of the system, without the need for any future information. It has been extensively used in routing and channel allocation in wireless networks [7][9], and has only recently been introduced to address resource allocation problems in a few other types of networks [10][11]. We tailor Lyapunov optimization techniques in the setting of a hybrid cloud, to dynamically and jointly resolve the optimal content replication and load distribution problems. We demonstrate the optimality of our algorithm with rigorous theoretical analysis. The algorithm nicely bounds the service response times within the preset QoS target in cases of arbitrary request arrivals, and guarantees that the overall cost is within a small constant gap from the optimum achieved by a T-slot lookahead mechanism with information into the future.

In the rest of the paper, we present the optimization model in

The research was supported in part by a grant from Hong Kong RGC under the contract HKU 718710E.

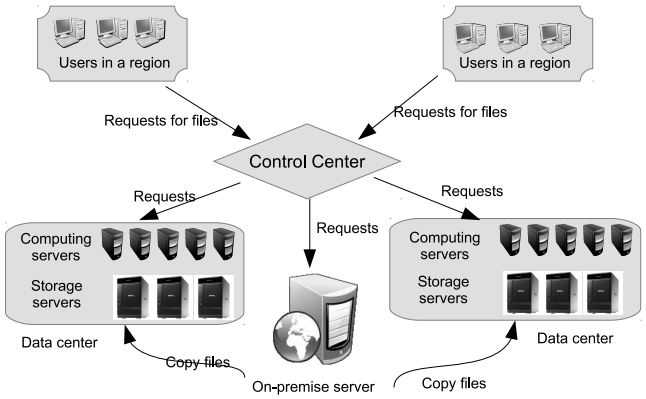

Fig. 1. The system architecture.

Sec. II, design a joint content placement and load distribution algorithm in Sec. III, analyze its performance in Sec. IV, and conclude the paper in Sec. V.

II. The Service Migration Problem

A. System Model

We consider a typical content distribution application, which provides a collection of contents (files), denoted as set $\mathcal{M}$, to users spreading over multiple geographical regions. There is an on-premise server cluster (or private cloud) owned by the provider of the content distribution application, which stores the original copies of all the contents. Without loss of generality, we use one server to represent the on-premise server cluster. The on-premise server has an overall upload bandwidth of b units for serving contents to users.

There is a public cloud consisting of data centers located in multiple geographical regions, denoted as set $\mathcal{N}$. One data center resides in each region. There are two types of inter-connected servers in each data center: storage servers for data storage, and computing servers that support the running and provisioning of virtual machines (VMs).

The provider of the content distribution application (*application provider*) wishes to provision its service by exploiting a *hybrid cloud* architecture, which includes the geo-distributed public cloud and its on-premise server. The major components of the content distribution application include: (i) back-end storage of the contents and (ii) front-end web service that serves users' requests for contents. The application provider may migrate both service components into the public cloud: contents can be replicated in storage servers in the cloud, while requests can be dispatched to web services installed on VMs on the computing servers. An illustration of the system architecture is given in Fig. 1.

We suppose that the system runs in a time-slotted fashion. Each time slot is a unit time which is enough for uploading any file $m \in \mathcal{M}$ with size $v^{(m)}$ (bytes) at the unit bandwidth. In time slot t, $a_j^{(m)}(t)$ requests are generated for downloading file $m \in \mathcal{M}$, from users in region j. We assume that the request arrival is an arbitrary process over time, and the number of requests arising from one region for a file in each time slot is upper-bounded by A_{max}.

TABLE I
Notations

$\mathcal{M}$	File set	$\mathcal{N}$	Region set
$v^{(m)}$	Size of file m, in bytes		
$a_j^{(m)}(t)$	No. of requests for file m from region j at time slot t		
$Q_j^{(m)}(t)$	Request queue for file m in region j		
A_{max}	Max. no. of requests for file m from region j in a time slot		
$s_j^{(m)}(t)$	No. of requests dispatched from $Q_j^{(m)}$ to on-premise server at t		
$c_{ji}^{(m)}(t)$	No. of requests dispatched from $Q_j^{(m)}$ to data center i at t		
$y_i^{(m)}(t)$	Binary var: store file m on data center i or not at t.		
b	Max. no. of requests the on-premise server can process in a time slot		
μ^{max}	Max. no. of requests dispatched from each request queue to a data center in a time slot, i.e., $c_{ji}^{(m)}(t) \leq \mu^{max}$		
g_i	Charge for uploading a byte from the data center i		
o_i	Charge for downloading a byte into data center i		
f_i	Charge for renting one VM instance in data center i		
r_i	No. of requests a VM in data center i can serve in a time slot		
p_i	Charge for storing a byte on data center i		
$q_i^{(m)}$	Charge for uploading file m from data center i		
h	Time-averaged charge for uploading a byte from the on-premise server		
$w_i^{(m)}$	Charge for migrating file m to data center i		
$W_j^{(m)}$	Bound of queueing delay of requests in queue $Q_j^{(m)}$		
$\epsilon_j^{(m)}$	Preset constant for controlling queueing delay in $Q_j^{(m)}$		
d_j	round-trip delay between region j and the on-premise server		
e_{ji}	round-trip delay between region j and data center i		
α	Bound of time-averaged round-trip delay		
$G(t)$	Virtual queue for bounding time-averaged round-trip delay		
$Z_j^{(m)}(t)$	Virtual queue for bounding queueing delay in $Q_j^{(m)}$		

The cost of uploading a byte from the on-premise server is h. The charge for storage at data center i is p_i per byte per unit time. g_i and o_i per byte are charged for uploading from and downloading into data center i, respectively. The cost for renting a VM instance in data center i is f_i per unit time. These charges follow the charging model of leading commercial cloud providers, such as Amazon EC2 [12] and S3[13]. We assume that the storage capacity in each data center is sufficient for storing contents from this content distribution application. We also assume that each request is served at one unit bandwidth, and the number of requests that a VM in data center i can serve per unit time is r_i.

B. Cost-Minimizing Service Migration Problem

We next develop an optimization framework to characterize the optimal content distribution service migration problem. Important notations are summarized in Table I.

The decision variables in our optimization framework are formulated as follows: (1) For *content replication*, binary variable $y_i^{(m)}(t)$ indicates whether file m is stored in data center i in time slot t or not. If $y_i^{(m)}(t-1) = 0$ and $y_i^{(m)}(t) = 1$, file m is copied from the on-premise server to the data center i at t; if $y_i^{(m)}(t-1) = 1$ and $y_i^{(m)}(t) = 0$, file m is removed from data center i. In other cases, the storage status remain the same. (2) For *dispatching requests* from region j for file m, let $s_j^{(m)}(t)$ be the number of requests to be served by the on-premise server in time slot t, and $c_{ji}^{(m)}(t)$ denote the

number dispatched to data center i in time slot t, with an upper bound of μ^{max}. Requests for file m can only be dispatched to data center i when it stores the file, *i.e.*, $c_{ji}^{(m)}(t) > 0$ only if $y_i^{(m)}(t) = 1$. We assume that a data center can serve a file to users in the time slot when the file is being copied to the data center, since replicating the file from the on-premise server and serving chunks of the file can be carried out in parallel.

To buffer requests for file m generated from users in region j over time, a queue $Q_j^{(m)}$ is maintained, $\forall j \in \mathcal{N}, m \in \mathcal{M}$. The backlog size of queue $Q_j^{(m)}$ at time t, denoted as $Q_j^{(m)}(t)$, is updated in each time slot as follows:

$$Q_j^{(m)}(t+1) = \max[Q_j^{(m)}(t) - s_j^{(m)}(t) - \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t), 0] + a_j^{(m)}(t). \tag{1}$$

Our framework focuses on minimizing the *recurring* operational cost in the content distribution system. Let $q_i^{(m)} = \frac{f_i}{r_i} + v^{(m)} g_i$ denote the unit cost to serve each request for file m on data center i and let $w_i^{(m)}$ denote the migration cost to copy file m into data center i, which includes costs of upload and download bandwidths from the on-premise server to data center i, *i.e.*, $w_i^{(m)} = v^{(m)}(h + o_i)$. The overall operational cost to the application provider in time slot t is

$$\begin{aligned} M(t) \quad = \quad & \sum_{m\in\mathcal{M}} \sum_{j\in\mathcal{N}} (v^{(m)} s_j^{(m)}(t) h + \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t) q_i^{(m)}) \\ & + \sum_{i\in\mathcal{N}} \sum_{m\in\mathcal{M}} v^{(m)} y_i^{(m)}(t) p_i \\ & + \sum_{i\in\mathcal{N}} \sum_{m\in\mathcal{M}} [y_i^{(m)}(t) - y_i^{(m)}(t-1)]^+ w_i^{(m)}, \end{aligned} \tag{2}$$

where the first row corresponds to the charge for uploading contents to users from the on-premise server and the cloud data centers, the second row computes the storage cost for caching replicated contents at all the data centers, and the third row is the migration cost for copying files from the on-premise server to the data centers. Here $[x]^+ = x$ if $x \geq 0$ and $[x]^+ = 0$ if $x < 0$. We will not consider any recurring storage cost on the on-premise server, and the removal of contents from a data center is cost-free.

On the other hand, our framework enforces pre-set QoS constraints. The service quality experienced by users is evaluated by request response delay, consisting of two major components: queueing delay in the request queue, and round-trip delay from when the request is dispatched from the queue to the time the first byte of the requested file is received. We ignore the processing delay inside a data center. Let d_j and e_{ji} denote the round-trip delay between region j and the on-premise server, and between region j and data center i, respectively. Let α be the upper-bound of the average round-trip delay per request, which the application provider wishes to enforce in this content distribution application. We reasonably assume $\alpha > e_{ii}, \forall i \in \mathcal{N}$, *i.e.*, this bound is larger than the round-trip delay between a user and the data center in the same region.

The optimization pursued by our dynamic algorithm is formulated as follows, which minimizes the time-averaged operational cost while guaranteeing service quality. We use $\overline{x(t)} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} x(t)$ to represent the time-averaged value of $x(t)$.

$$\min \overline{M(t)} \tag{3}$$

subject to:

$$\sum_{m\in\mathcal{M}} \sum_{j\in\mathcal{N}} s_j^{(m)}(t) \leq b, \forall t, \tag{4}$$

$$0 \leq c_{ji}^{(m)}(t) \leq \mu_{max} y_i^{(m)}(t), \forall j \in \mathcal{N}, i \in \mathcal{N}, m \in \mathcal{M}, \forall t, \tag{5}$$

$$s_j^{(m)}(t) + \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t) \leq Q_j^{(m)}(t), \forall m \in \mathcal{M}, j \in \mathcal{N}, \forall t, \tag{6}$$

$$\overline{\sum_{j\in\mathcal{N}} \sum_{m\in\mathcal{M}} (s_j^{(m)}(t) d_j + \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t) e_{ji})} < \alpha \overline{\sum_{j\in\mathcal{N}} \sum_{m\in\mathcal{M}} (s_j^{(m)}(t) + \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t))}, \tag{7}$$

$$s_j^{(m)}(t) \geq 0, \forall j \in \mathcal{N}, m \in \mathcal{M}, \forall t, \tag{8}$$

$$y_i^{(m)}(t) \in \{0, 1\}, \forall i \in \mathcal{N}, m \in \mathcal{M}, \forall t. \tag{9}$$

(4) corresponds to the upload bandwidth limit at the on-premise server. (5) states that requests for a file are only dispatched to data centers storing this file, and the maximum number of requests dispatched from each request queue to a data center in each time slot is no larger than μ^{max}. (6) states that the number of requests dispatched from queue $Q_j^{(m)}$ cannot be larger than the current queue size. Constraint (7) specifies that the average round-trip delay per request should be bounded by α.

Though queueing delay is not explicitly modeled in the constraints, we show in Sec. IV that our algorithm can simultaneously solve this optimization and bound the queueing delay of each request with a pre-set threshold.

III. Dynamic Migration Algorithm

We next design a dynamic control algorithm using Lyapunov optimization techniques, which solves the optimal migration problem in (3) and bounds the time-averaged round-trip delays and queueing delays for each request.

A. Bounding Delays

To satisfy constraint (7), we resort to the virtual queue techniques in Lyapunov optimization [8]. We introduce a virtual queue G, with arrival rate of $\sum_{j\in\mathcal{N}} \sum_{m\in\mathcal{M}} (s_j^{(m)}(t) d_j + \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t) e_{ji})$, *i.e.*, the overall round-trip delay experienced by all requests in t, and departure rate of $\alpha \sum_{j\in\mathcal{N}} \sum_{m\in\mathcal{M}} (s_j^{(m)}(t) + \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t))$, *i.e.*, the total number of requests in t multiplied by the round-trip delay bound α. G is updated as follows:

$$\begin{aligned} G(t+1) = \max[G(t) + & \sum_{j\in\mathcal{N}} \sum_{m\in\mathcal{M}} (s_j^{(m)}(t) d_j + \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t) e_{ji}) \\ & - \alpha \sum_{j\in\mathcal{N}} \sum_{m\in\mathcal{M}} (s_j^{(m)}(t) + \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t)), 0]. \end{aligned} \tag{10}$$

If queue G is stable, its time-averaged arrival rate should not exceed its time-averaged departure rate [8], *i.e.*, constraint (7) is satisfied. Therefore, we can adjust the request distribution strategies $s_j^{(m)}(t)$'s and $c_{ji}^{(m)}(t)$'s in each time slot to guarantee that the virtual queue is always stable, in order to satisfy constraint (7).

To bound the worst-case queueing delay of each request in all queues $Q_j^{(m)}, \forall j \in \mathcal{N}, m \in \mathcal{M}$, the *$\epsilon$-persistent service queue* technique [14] can be applied. In particular, we associate $Q_j^{(m)}$ with a virtual queue $Z_j^{(m)}$, updated by:

$$Z_j^{(m)}(t+1) = \max[Z_j^{(m)}(t) + 1_{\{Q_j^{(m)}(t)>0\}}(\epsilon_j^{(m)} - s_j^{(m)}(t) - \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t)) - 1_{\{Q_j^{(m)}(t)=0\}}\mu_{max}, 0], \quad (11)$$

where $\epsilon_j^{(m)} > 0$ is a constant that can be gauged to control the queueing delay bound, which further renders a tradeoff between the queueing delay bound and the cost optimality achieved by our algorithm (to be discussed in Sec. IV). The rationale behind (11) can be explained intuitively: If request queue $Q_j^{(m)}$ is not empty in time slot t, then a constant number of arrivals $\epsilon_j^{(m)}$ are added into virtual queue $Z_j^{(m)}$, while the departure rate from the virtual queue, $s_j^{(m)}(t)+\sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t)$, is the same as the departure rate from request queue $Q_j^{(m)}$. If the request queue is empty in t, the length of the virtual queue decreases by μ_{max}. We strategically decide $s_j^{(m)}(t)$'s and $c_{ji}^{(m)}(t)$'s to keep the virtual queue stable; in this way, requests are expediently dispatched from the request queue, resulting in limited queueing delay per request.

B. Dynamic Algorithm Design

Next we design a dynamic algorithm which stabilizes all queues and solves optimization (3).

Let $\mathbf{\Theta}(t) = [\mathbf{Q}(t), \mathbf{G}(t), \mathbf{Z}(t)]$ be the vector of all queues in the system. Define our Lyapunov function as

$$L(\mathbf{\Theta}(t)) = \frac{1}{2}[\sum_{m\in\mathcal{M}}\sum_{j\in\mathcal{N}}(Q_j^{(m)}(t)^2 + Z_j^{(m)}(t))^2 + G(t)^2]. \quad (12)$$

The one-slot conditional Lyapunov drift is

$$\Delta(\mathbf{\Theta}(t)) = \mathbb{E}\{L(\mathbf{\Theta}(t+1)) - L(\mathbf{\Theta}(t))|\mathbf{\Theta}(t)\}.$$

Following the *drift-plus-penalty* framework in Lyapunov optimization (Chapter 5 in [8]), we can minimize the time-averaged operational cost and stabilize all queues by minimizing an upper bound of the following item in each time slot:

$$\Delta(\mathbf{\Theta}(t)) + VM(t),$$

where V is a non-negative parameter set by the application provider to control the tradeoff between the operational cost and request response delays.

Squaring the queueing laws (1), (10) and (11), we derive the following inequality:

$$\begin{aligned}
&\Delta(\mathbf{\Theta}(t)) + VM(t)\\
&\le B - \sum_{m\in\mathcal{M}}\sum_{j\in\mathcal{N}} s_j^{(m)}(t)[Q_j^{(m)}(t) + (\alpha - d_j)G(t)\\
&\quad + 1_{\{Q_j^{(m)}(t)>0\}}Z_j^{(m)}(t) - v^{(m)}Vh] - \sum_{m\in\mathcal{M}}\sum_{j\in\mathcal{N}}\sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t)\\
&\quad [Q_j^{(m)}(t) + (\alpha - e_{ji})G(t) + 1_{\{Q_j^{(m)}(t)>0\}}Z_j^{(m)}(t) - Vq_i^{(m)}]\\
&\quad + V\sum_{m\in\mathcal{M}}\sum_{i\in\mathcal{N}}[v^{(m)}y_i^{(m)}(t)p_i + [y_i^{(m)}(t) - y_i^{(m)}(t-1)]^+ w_i^{(m)}]\\
&\quad + \sum_{m\in\mathcal{M}}\sum_{j\in\mathcal{N}} Z_j^{(m)}(t)[1_{\{Q_j^{(m)}(t)>0\}}\epsilon_j^{(m)} - 1_{\{Q_j^{(m)}=0\}}\mu_{max}]\\
&\quad + \sum_{m\in\mathcal{M}}\sum_{j\in\mathcal{N}} Q_j^{(m)}(t)a_j^{(m)}(t),
\end{aligned} \quad (13)$$

Algorithm 1: Control Algorithm on the Control Center

Initialization:
Set up request queue $Q_j^{(m)}$, virtual queues G and $Z_j^{(m)}, \forall j \in \mathcal{N}, m \in \mathcal{M}$, and initialize their backlogs to 0;
In every time slot t:
1. Enqueue received requests to request queues ($Q_j^{(m)}$'s);
2. Solve optimization (14) to obtain optimal content placement and load distribution strategies $c_{ji}^{(m)}(t)$, $s_j^{(m)}(t)$, $y_i^{(m)}(t)$, $\forall j, i \in \mathcal{N}, m \in \mathcal{M}$;
3. Update content placement table with $y_i^{(m)}(t)$'s, and migrate files as follows:
for $i \in \mathcal{N}, m \in \mathcal{M}$ **do**
  if $y_j^{(m)}(t-1) = 0$ *and* $y_j^{(m)}(t) = 1$ **then**
    instruct on-premise server to upload file m to data center i;
  if $y_j^{(m)}(t-1) = 1$ *and* $y_j^{(m)}(t) = 0$ **then**
    signal data center i to remove file m;
4. Dispatch $s_j^{(m)}(t)$ requests from queue $Q_j^{(m)}$ to on-premise server, $c_{ji}^{(m)}(t)$ requests to data center i, $\forall j, i \in \mathcal{N}, m \in \mathcal{M}$;
5. Update virtual queue $Z_{ji}^{(m)}$ and G according to Eqn. (11) and (10);

where $B = \frac{1}{2}|\mathcal{M}||\mathcal{N}|[A_{max}^2 + \epsilon_{max}^2 + 2(b + N\mu_{max})^2] + +\frac{1}{2}(|\mathcal{M}||\mathcal{N}|^2\mu_{max}e_{max} + bd_{max})^2 + \frac{1}{2}\alpha^2(|\mathcal{M}||\mathcal{N}|^2\mu_{max} + b)^2$ is a constant, with $d_{max} = \max\{d_j | j \in \mathcal{N}\}$, $e_{max} = \max\{e_{ji} | j \in \mathcal{N}, i \in \mathcal{N}\}$, and $\epsilon_{max} = \max\{\epsilon_j^{(m)} | j \in \mathcal{N}, m \in \mathcal{M}\}$. We simplify the notation by defining

$$\gamma_j^{(m)}(t) = Q_j^{(m)}(t) + 1_{\{Q_j^{(m)}(t)>0\}}Z_j^{(m)}(t) - Vv^{(m)}h + (\alpha - d_j)G(t),$$

which is a constant in time slot t, and

$$\eta_{ji}^{(m)}(t) = Q_j^{(m)}(t) + 1_{\{Q_j^{(m)}(t)>0\}}Z_j^{(m)}(t) - Vq_i^{(m)} + (\alpha - e_{ji})G(t),$$

which is also a constant in t, and

$$\phi_i^{(m)}(t) = V(v^{(m)}p_i + 1_{\{y_i^{(m)}(t-1)=0\}}w_i^{(m)}),$$

which is a constant in t as well, when $y_i^{(m)}(t-1)$ is given.

Minimizing the right-hand-side of (13) is equivalent to:

$$\begin{aligned}
\max \quad & F(t) = \sum_{m\in\mathcal{M}}\sum_{j\in\mathcal{N}} s_j^{(m)}(t)\gamma_j^{(m)}(t) +\\
& \sum_{m\in\mathcal{M}}\sum_{j\in\mathcal{N}}\sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t)\eta_{ji}^{(m)}(t) - \sum_{m\in\mathcal{M}}\sum_{i\in\mathcal{N}}\phi_i^{(m)}(t)y_i^{(m)}(t) \quad (14)\\
& \text{subject to: constraints (4) (6) (5) (8) (9).}
\end{aligned}$$

This problem is an integer linear program (ILP), which can be solved by optimization tools such as *GLPK* [15].

In summary, our complete dynamic, joint content placement and request distribution algorithm is given in Algorithm 1. The algorithm can be implemented by the application provider on a *control center*. The control center maintains a content placement table with entries $y_i^{(m)}$, which are initialized to be 0. In each time slot, it receives user requests and places the requests for file m originated from region j in request queue $Q_j^{(m)}$. Virtual queues $Z_j^{(m)}$ and G are maintained simply as counters. The control center observes the lengths of the queues and request arrival rates, and calculates the optimal content placement and load distribution strategies using Algorithm 1. It then signals the cloud data centers to replicate/remove files and dispatches requests across the hybrid cloud accordingly.

IV. Performance Analysis

We next analyze the performance guarantee provided by our dynamic algorithm. Detailed proofs to all the theorems can be found in our technical report [16].

A. Bound of Queueing Delay

Theorem 1: (Bound of Queue Length) Define

$$Q_j^{(m)^{max}} = V(v^{(m)}p_{\tilde{i}} + w_{\tilde{i}}^{(m)} + q_{\tilde{i}}^{(m)}) + A_{max}, \text{where} \quad (15)$$

$$\tilde{i} = \text{argmin}_i\{v^{(m)}p_i + w_i^{(m)} + q_i^{(m)} | \alpha - e_{ji} > 0, \forall i \in \mathcal{N}\}. \quad (16)$$

Then $Q_j^{(m)^{max}}$ is the maximum size of queue $Q_j^{(m)}$ at any time t, *i.e.*, $Q_j^{(m)}(t) \leq Q_j^{(m)^{max}}$, $\forall j \in \mathcal{N}, \forall m \in \mathcal{M}$.

Theorem 2: (Bounded Queueing Delay): For each request queue $Q_j^{(m)}$, $\forall j \in \mathcal{N}, \forall m \in \mathcal{M}$, define

$$W_j^{(m)} = \lceil \frac{V(v^{(m)}p_{\tilde{i}} + w_{\tilde{i}}^{(m)} + q_{\tilde{i}}^{(m)}) + Q_j^{(m)^{max}}}{\epsilon_m} \rceil$$

where $\tilde{i}$ is defined as in (16). The queueing delay of each request in $Q_j^{(m)}$ is bounded by $W_j^{(m)}$.

B. Optimality against the T-Slot Lookahead Mechanism

Since request arrival rates are arbitrary in our system, it is difficult to find the global cost optimum, with which to compare the time-averaged cost $\overline{M(t)}$ achieved by our algorithm. Therefore we utilize a local optimum target, which is the optimal (objective function) value of a similar cost minimization problem within known information (*e.g.*, request arrivals) for T time slots into the future, *i.e.*, a *T-slot lookahead* mechanism [8]. In the T-slot lookahead mechanism, time is divided into successive frames, each consisting of T time slots. Denote each frame as $F_k = \{kT, kT+1, \ldots, kT+T-1\}$, where $k = 0, 1, \ldots$. In each time frame, consider the following optimization problem on variables $c_{ji}^{(m)}(t), s_j^{(m)}(t), y_i^{(m)}(t)$, $\forall j \in \mathcal{N}, i \in \mathcal{N}, m \in \mathcal{M}$, $t \in F_k$:

$$\min \frac{1}{T} \sum_{t=kT}^{kT+T-1} M(t) \quad (17)$$

subject to:

$$\sum_{m\in\mathcal{M}} \sum_{j\in\mathcal{N}} s_j^{(m)}(t) \leq b, \forall t \in F_k,$$

$$0 \leq c_{ji}^{(m)}(t) \leq \mu_{max} y_i^{(m)}(t), \forall j \in \mathcal{N}, \forall i \in \mathcal{N}, m \in \mathcal{M}, t \in F_k,$$

$$s_j^{(m)}(t) + \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t) \leq Q_j^{(m)}(t), \forall m \in \mathcal{M}, \forall j \in \mathcal{N}, \forall t \in F_k,$$

$$\sum_{t=kT}^{kT+T-1} [a_j^{(m)}(t) - s_j^{(m)}(t) - \sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t)] \leq 0, \forall j \in \mathcal{N}, m \in \mathcal{M},$$

$$\sum_{t=kT}^{kT+T-1} \sum_{j\in\mathcal{N}} \sum_{m\in\mathcal{M}} (\sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t) e_{ji} + s_j^{(m)}(t) d_j) < \alpha \sum_{t=kT}^{kT+T-1} \sum_{j\in\mathcal{N}} \sum_{m\in\mathcal{M}} (\sum_{i\in\mathcal{N}} c_{ji}^{(m)}(t) + s_j^{(m)}(t)),$$

$$s_j^{(m)}(t) \geq 0, \forall j \in \mathcal{N}, m \in \mathcal{M}, t \in F_k,$$

$$y_i^{(m)}(t) \in \{0, 1\}, \forall i \in \mathcal{N}, m \in \mathcal{M}, t \in F_k.$$

Theorem 3: (Optimality of Cost) Let $\widehat{M_k}$ denote the optimal objective function value in the T-slot Lookahead problem (17) in time frame F_k. The minimum operational cost derived with our algorithm is $M(t)$ in time slot t. Suppose the time lasts for KT time slots, where K is a constant. We have

$$\frac{1}{KT} \sum_{t=0}^{KT-1} M(t) \leq \frac{1}{K} \sum_{k=0}^{K-1} \widehat{M_k} + \frac{BT}{V}, \quad (18)$$

i.e., our algorithm achieves a time-averaged cost within constant gap $\frac{BT}{V}$ from that by assuming full knowledge in T time slots in the future.

Theorems 2 and 3 show that when V increases, worst-case queueing delay $W_j^{(m)}$ increases, while the gap between the operational cost of our algorithm and that of the T-Slot lookahead mechanism is reduced. $\epsilon_j^{(m)}$ has a similar effect: when $\epsilon_j^{(m)}$ increases, the worst-case queueing delay $W_j^{(m)}$ decreases, and B increases such that the gap to optimality increases.

V. Conclusion

This paper investigates optimal migration of a content distribution service to a hybrid cloud consisting of private on-premise servers and public geo-distributed cloud services. We propose a generic optimization framework based on Lyapunov optimization theory, and design a dynamic, joint content placement and request distribution algorithm, which minimizes the operational cost of the application with QoS guarantees. We theoretically show that our algorithm approaches the optimality achieved by a mechanism with known information in the future T time slots by a small constant gap, no matter what the request arrival pattern is. We intend to extend the framework to specific content distribution services with detailed requirements, such as video-on-demand services or social media applications, in our ongoing work.

References

[1] *Microsoft Office Web Apps*, http://office.microsoft.com/en-us/web-apps/.

[2] *Google docs*, http://docs.google.com/.

[3] M. Hajjat, X. Sun, Y. E. Sung, D. Maltz, and S. Rao, "Cloudward Bound: Planning for Beneficial Migration of Enterprise Applications to the Cloud," in *Proc. of IEEE SIGCOMM*, August 2010.

[4] H. Zhang, G. Jiang, K. Yoshihira, H. Chen, and A. Saxena, "Intelligent Workload Factoring for a Hybrid Cloud Computing Model," in *Proc. of the International Workshop on Cloud Services (IWCS 2009)*, June 2009.

[5] H. Li, L. Zhong, J. Liu, B. Li, and K. Xu, "Cost-effective Partial Migration of VoD Services to Content Clouds," in *Proc. of IEEE CLOUD*, July 2011.

[6] X. Cheng and J. Liu, "Load-Balanced Migration of Social Media to Content Clouds," in *Proc. of NOSSDAV*, June 2011.

[7] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1–149, 2006.

[8] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool, 2010.

[9] M. J. Neely, "Energy optimal control for time varying wireless networks," *IEEE Tran. on Information Theory*, no. 7, pp. 2915–2934, July 2006.

[10] M. M. Amble, P. Parag, S. Shakkottai, and L. Ying, "Content-Aware Caching and Traffic Management in Content Distribution Networks," in *Proc. of IEEE INFOCOM*, April 2011.

[11] M. J. Neely and L. Golubchik, "Utility Optimization for Dynamic Peer-to-Peer Networks with Tit-For-Tat Constraints," in *Proc. of IEEE INFOCOM*, April 2011.

[12] *Amazon Elastic Compute Cloud*, http://aws.amazon.com/ec2/.

[13] *Amazon Simple Storage Service*, http://aws.amazon.com/s3/.

[14] M. J. Neely, "Opportunistic Scheduling with Worst Case Delay Guarantees in Single and Multi-Hop Networks," in *Proc. of IEEE INFOCOM*, 2011.

[15] *GLPK (GNU Linear Programming Kit)*, http://www.gnu.org/s/glpk/.

[16] X. Qiu, H. Li, C. Wu, Z. Li, and F. C. M. Lau, "Cost-Minimizing Dynamic Migration of Content Distribution Services into Hybrid Clouds," `http://www.cs.hku.hk/~xjqiu/xjqiu-infocom12.pdf`, The University of Hong Kong, Tech. Rep., Jan. 2012.