Cost Efficient Design of Fault Tolerant Geo-Distributed Data Centers

Rakesh Tripathi, S. Vignesh, Venkatesh Tamarapalli and Deep Medhi

Abstract—Many critical e-commerce and financial services are deployed on geo-distributed data centers for scalability and availability. Recent market surveys show that failure of a data center is inevitable resulting in a huge financial loss. Fault-tolerance in distributed data centers is typically handled by provisioning spare capacity to mask failure at a site. We argue that the operating cost and data replication cost (for data availability) must be considered in spare capacity provisioning along with minimizing the number of servers. Since the operating cost and client demand vary across space and time, we propose cost-aware capacity provisioning to minimize the total cost of ownership (TCO) for fault-tolerant data centers.

We formulate the problem of spare capacity provisioning in fault-tolerant distributed data centers using mixed integer linear programming (MILP), with an objective of minimizing the TCO. The model accounts for heterogeneous client demand, data replication strategies (single and multiple site), variation in electricity price and carbon tax, and delay constraints while computing the spare capacity. Solving the MILP using real-world data, we observed a saving in the TCO to the tune of 35% compared to the model that minimizes the total number of servers and 43% compared to the model that minimizes the average response time. We demonstrate that our model is beneficial when the cost of electricity, carbon tax, and bandwidth vary significantly across the locations, which seems to be the problem for most of the operators.

Index Terms—Geo-distributed data center, capacity provisioning, fault tolerance, mixed integer linear programming

I. INTRODUCTION

Recently, a number of Internet services and applications were deployed over large scale geo-distributed data centers. A geo-distributed data center is an orchestrated collection of data centers, distributed across several locations and transparently interconnected with overlay links [1]. Geo-distributed data centers offer advantages such as increased availability, lower access time for users across the globe and horizontal scale out against capacity constraints (electricity, physical space, etc.). Due to these advantages, several cloud providers like Amazon and content distribution companies such as Akamai, invest in building geo-distributed data centers. For example, Google has data centers across 15 countries at more than 30 sites with an estimated 900,000 servers [2].

Critical e-commerce and financial services running on geo-distributed data centers (henceforth simply referred to as data centers) demand high availability because of a huge loss of revenue associated with downtime. A survey by Gartner estimated that 60% of companies incurred a loss to the tune of $250,000-$500,000 for an hour of downtime, and one sixth of the companies incurred a loss of $1 million or more [3]. Further, the latest survey by Ponemon Institute showed that the frequency of data center outage (complete or partial) could be as high as once a month with an average duration of three hours. It was reported to cause a loss of $1,734,433 per organization with an average cost of $690,204 per incident [4]. Instances of a data center failure at a site have been reported by many cloud service providers like Amazon, Facebook, and Google [5], [6]. These failures are attributed to various reasons like power outages, cable cuts, software bugs, mis-configured routers, DDoS attacks, and natural disasters [7]. In this paper, by high availability we mean that the data center continues to deliver original service (may be with a degraded performance) after failure of a single site. This can be achieved by providing spare compute capacity across the data centers.

Along with service restoration, it is also important that the required data is available at an alternate location after failure. This is handled by replication of data according to a predetermined policy. There are two options possible for data replication namely, single site replication and multiple site replication. In single site replication, the data is replicated to another nearby data center. In case of a failure, if the replicated site is overloaded, client requests are directed to any other data center meeting the latency requirement. In this case, the data would be pulled from the replica, which results in greater latency and bandwidth cost (we call this a post-failure penalty). In order to ensure co-location of data with the compute servers, the data is often replicated at multiple sites. However, multi-site replication involves large replication cost since the data center operators are typically charged for the number of bytes transferred [8] and/or the bandwidth cost between the replication sites [9]. Therefore, the replication cost should be considered while designing the data centers for high availability.

In summary, designing a fault-tolerant, highly available, distributed data center involves minimizing the spare capacity (number of servers) across the data centers considering the cost of power consumed and data replication, subject to a set of constraints related to client demand, delay bound, and the power and capacity available. We call this problem cost-aware capacity provisioning (CACP) wherein, the main challenge is to minimize the total cost of ownership (TCO) for data center operators by leveraging the spatio-temporal variation in electricity price and user demand.

Manuscript received September 15, 2016; revised January 23, 2017. The associate editor coordinating the review of this paper and approving it for publication was Yixin Diao.

Rakesh Tripathi, S. Vignesh, and Venkatesh Tamarapalli are with Department of CSE, Indian Institute of Technology Guwahati, Assam, India. E-mail: {trakesh, s.vignesh, t.venkat}@iitg.ernet.in

Deep Medhi is with Department of CSEE, University of Missouri - Kansas City, USA. E-mail: dmedhi@umkc.edu
We proved that the CACP problem for the design of a fault-tolerant distributed data center is NP-hard.

Replication Cost: Usually, cloud service providers connect their data centers with dedicated WAN links that are significantly expensive. Therefore, informed data replication must be carried out in order to minimize the operating cost involved. For example, AWS charges an inter-data center transfer, around $0.12-0.2/GB across geographic regions and $0.01/GB in the same region [8]. Literature also suggests that the data replication may be charged based on the distance between the replicating sites, e.g., $1 to transfer 2.7 GB of data over 100km was reported in [9].

In this paper, we give an MILP-based solution for the CACP problem to optimize the TCO, while complying to the customer demand, latency requirements, and being cost effective while masking the failure of any one data center. We summarize the main contributions as follows:

- We formulated the CACP problem as a mixed integer linear program with an objective of minimizing the TCO (includes cost of server acquisition, electricity, carbon tax, and data replication) subject to latency, power, demand, and availability constraints.
- We proved that the CACP problem for the design of a fault-tolerant distributed data center is NP-hard.
- We collected traces from Wikipedia.org [15] sites to create a heterogeneous workload and used it to model the server utilization under heterogeneous demand.
- We modelled two strategies for data replication, single site and multiple site for data affinity. Evaluation of these models with our framework suggests that although the multiple site model is costlier, it is preferable when the post-failure penalty is large in the single site model.
- We used real-world data for the price of electricity to evaluate the proposed model that shows the CACP model results in significant savings in the TCO compared to the existing models.

The rest of the paper is organized as follows. Section II discusses the work related to capacity provisioning in distributed data centers. Section III presents the cost models used, formulation of the CACP problem, and discusses the complexity of the formulation. We also illustrate the working of the model with a small example. Results demonstrating the advantages of the proposed model over the existing ones are reported in Section IV. Section V concludes the paper.

### II. Related Work

There have been significant efforts to address the problems of server consolidation, server switching, VM migration, and load balancing to minimize the operating cost under the assumption that sufficient servers are already provisioned (for e.g., [16], [17]). However, there is not much work done in planning data center deployment considering failures and the offline problem of capacity allocation to design fault-tolerant data centers. This section discusses the literature addressing server placement and capacity provisioning in geo-distributed data centers.

The authors of [9] proposed an optimization framework to provision servers across different locations using three different objectives: to minimize the total carbon footprint, to minimize the total cost, and to minimize average service latency. The costs considered were the electricity cost and bandwidth cost, with the constraints related to client latency. In [10], the authors proposed a mechanism to select data center locations to minimize the total cost of ownership that includes capital as well as operating costs subject to delay, consistency, and availability constraints. The capital cost factors included the cost of land, data center construction, transmission line to power grid, OFC line to network backbone, cooling infrastructure, and internal network. The operating cost factors included cost of electricity, bandwidth, cooling the data center, carbon tax, and administration.

The work in [18] addressed the problem of maximizing profit by either building a new data center or by expanding the existing ones (increase the number of servers) to meet the increasing demand. Along with the cost of electricity, cooling, bandwidth and revenue generated, the annual inflation rate over a period of time was also included in the profit generated. The MILP optimization framework determines the best option to maximize revenue for a given data center location and its compute capacity. The work in [19] jointly handled the data center server placement, capacity provisioning and request

### Table I: EAC for different countries

<table>
<thead>
<tr>
<th>Country/Area</th>
<th>Electricity price($/kWh)</th>
<th>Cost(in $)</th>
<th>EAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canada</td>
<td>0.06</td>
<td>946</td>
<td>47</td>
</tr>
<tr>
<td>Oregon, USA</td>
<td>0.06</td>
<td>946</td>
<td>47</td>
</tr>
<tr>
<td>Virginia, USA</td>
<td>0.07</td>
<td>1104</td>
<td>55</td>
</tr>
<tr>
<td>Switzerland</td>
<td>0.07</td>
<td>1230</td>
<td>62</td>
</tr>
<tr>
<td>Netherlands</td>
<td>0.09</td>
<td>1419</td>
<td>71</td>
</tr>
<tr>
<td>Japan</td>
<td>0.10</td>
<td>17</td>
<td>84</td>
</tr>
<tr>
<td>California, USA</td>
<td>0.12</td>
<td>1971</td>
<td>99</td>
</tr>
<tr>
<td>Ireland</td>
<td>0.13</td>
<td>2050</td>
<td>103</td>
</tr>
<tr>
<td>UK</td>
<td>0.13</td>
<td>2050</td>
<td>103</td>
</tr>
<tr>
<td>Hongkong</td>
<td>0.17</td>
<td>2680</td>
<td>134</td>
</tr>
</tbody>
</table>
We assumed that the failure of the data center at a site is

The demand from a client region is proportional to the

Failure detection and request re-routing is handled by the

Data replication happens with any popular geo-distributed

Therefore, we used minimization of the TCO as an objective

A. Assumptions

The following assumptions were made in the model.

• We assumed that the failure of the data center at a site is

• Data replication happens with any popular geo-distributed

• Failure detection and request re-routing is handled by the

• All the servers have similar configurations and can serve

B. System Model

In this section, we define the variables and cost models

<table>
<thead>
<tr>
<th>Variable Parameters</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td>set of data center locations</td>
</tr>
<tr>
<td>$U$</td>
<td>set of client locations</td>
</tr>
<tr>
<td>$A$</td>
<td>set of application types</td>
</tr>
<tr>
<td>$H$</td>
<td>total time horizon</td>
</tr>
<tr>
<td>$s$</td>
<td>index for data center location</td>
</tr>
<tr>
<td>$u$</td>
<td>index for client region</td>
</tr>
<tr>
<td>$f$</td>
<td>index for failed data center</td>
</tr>
<tr>
<td>$h$</td>
<td>index for time slot in time horizon</td>
</tr>
<tr>
<td>$a$</td>
<td>index for application request type</td>
</tr>
<tr>
<td>$B$</td>
<td>processing rate of server in bits per second</td>
</tr>
<tr>
<td>$J_a$</td>
<td>job size for request of type $a$ in kB</td>
</tr>
<tr>
<td>$p_s^h$</td>
<td>power consumed at data center $s$ for application $a$ during hour $h$ with failed data center $f$</td>
</tr>
<tr>
<td>$p_s^{max}$</td>
<td>maximum power available at data center $s$ during hour $h$</td>
</tr>
<tr>
<td>$\gamma^h_s$</td>
<td>average server utilization at data center $s$ during hour $h$ and failed data center $f$</td>
</tr>
<tr>
<td>$\gamma^{max}_s$</td>
<td>maximum value of $\gamma$ to avoid waiting</td>
</tr>
<tr>
<td>$L^h_u$</td>
<td>total number of requests generated for application $A$ from user location $u$ during hour $h$</td>
</tr>
<tr>
<td>$D_{su}$</td>
<td>propagation delay between client region $u$ and data center $s$</td>
</tr>
<tr>
<td>$D_{max}$</td>
<td>the maximum tolerable latency</td>
</tr>
<tr>
<td>$\Phi_s$</td>
<td>electricity price per KWh at data center $s$ at hour $h$</td>
</tr>
<tr>
<td>$\rho_s$</td>
<td>transmission loss of electricity at data center $s$</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>server acquisition cost</td>
</tr>
<tr>
<td>$\delta_s$</td>
<td>carbon tax at data center $s$</td>
</tr>
<tr>
<td>$M^{min}_s$</td>
<td>minimum number of servers at any data center</td>
</tr>
<tr>
<td>$M^{max}_s$</td>
<td>maximum number of servers at any data center</td>
</tr>
<tr>
<td>$v_{su}$</td>
<td>bandwidth cost for data center $s$ to data center $i$</td>
</tr>
<tr>
<td>$\xi_s$</td>
<td>number of bytes required for data replication of single request</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decision Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m_s$</td>
</tr>
<tr>
<td>$\lambda^{fh}_{su}$</td>
</tr>
<tr>
<td>$y_{su}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cost Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I$</td>
</tr>
<tr>
<td>$\Phi$</td>
</tr>
<tr>
<td>$\eta$</td>
</tr>
<tr>
<td>$\kappa$</td>
</tr>
<tr>
<td>$\Theta$</td>
</tr>
<tr>
<td>$\tau$</td>
</tr>
</tbody>
</table>

### III. MILP Model Formulation

In this section, we first state the assumptions used in the model and present the models considered for various cost factors. Next, we present the MILP formulation of the CACP problem and also prove that the problem is NP hard.

#### A. Assumptions

The following assumptions were made in the model.

• We assumed that the failure of the data center at a site is an independent process, i.e., data centers are not susceptible to common disaster situations [20]. For example, a power outage, building fire or any local disaster at one data center location will not effect the remaining data centers.

• Data replication happens with any popular geo-distributed data replication strategy.

• Failure detection and request re-routing is handled by the front-end proxy.

• Data centers are connected using dedicated virtual links and the cost of the data transfer is based on the actual usage.

• The demand from a client region is proportional to the population. Propagation delay within the client region is assumed to be negligible.

• All the servers have similar configurations and can serve requests for any service. However, the response sizes can be variable.

#### B. System Model

In this section, we define the variables and cost models used in the formulation. Table II lists all the input parameters, variables, and cost factors in the model.

### Failures: Let $S$ denote the set of data centers. The data centers are indexed between 1 and $|S|$. We use an index variable $f$ to represent the failure of a data center. $f$ takes values from the set $\{0,|S|\}$, where $f = 0$ indicates the case of no failure and $f = s$ indicates that the data center indexed $s \in \{1,2,\ldots,|S|\}$ has failed. We assume that the probability of a single data center failure, i.e., $f \neq 0$, is very small.

### Demand: Let $\lambda^{fh}_{su}$ denote the number of requests for an application type $a$, from a client region $u$, served by the data center at site $s$, during hour $h$ after the data center indexed $f \in \{1,|S|\}$ has failed. Let $L^h_u$ be the total demand from the client region $u$ at hour $h$.

### Server Provisioning: Let $m_s$ denote the number of servers required in a data center at $s$. We define $M^{min}_s$ and $M^{max}_s$ to be the minimum and maximum number of servers that can be provisioned at any data center based on the space and power

<table>
<thead>
<tr>
<th>Variable Parameters</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td>set of data center locations</td>
</tr>
<tr>
<td>$U$</td>
<td>set of client locations</td>
</tr>
<tr>
<td>$A$</td>
<td>set of application types</td>
</tr>
<tr>
<td>$H$</td>
<td>total time horizon</td>
</tr>
<tr>
<td>$s$</td>
<td>index for data center location</td>
</tr>
<tr>
<td>$u$</td>
<td>index for client region</td>
</tr>
<tr>
<td>$f$</td>
<td>index for failed data center</td>
</tr>
<tr>
<td>$h$</td>
<td>index for time slot in time horizon</td>
</tr>
<tr>
<td>$a$</td>
<td>index for application request type</td>
</tr>
<tr>
<td>$B$</td>
<td>processing rate of server in bits per second</td>
</tr>
<tr>
<td>$J_a$</td>
<td>job size for request of type $a$ in kB</td>
</tr>
<tr>
<td>$p_s^h$</td>
<td>power consumed at data center $s$ for application $a$ during hour $h$ with failed data center $f$</td>
</tr>
<tr>
<td>$p_s^{max}$</td>
<td>maximum power available at data center $s$ during hour $h$</td>
</tr>
<tr>
<td>$\gamma^h_s$</td>
<td>average server utilization at data center $s$ during hour $h$ and failed data center $f$</td>
</tr>
<tr>
<td>$\gamma^{max}_s$</td>
<td>maximum value of $\gamma$ to avoid waiting</td>
</tr>
<tr>
<td>$L^h_u$</td>
<td>total number of requests generated for application $A$ from user location $u$ during hour $h$</td>
</tr>
<tr>
<td>$D_{su}$</td>
<td>propagation delay between client region $u$ and data center $s$</td>
</tr>
<tr>
<td>$D_{max}$</td>
<td>the maximum tolerable latency</td>
</tr>
<tr>
<td>$\Phi_s$</td>
<td>electricity price per KWh at data center $s$ at hour $h$</td>
</tr>
<tr>
<td>$\rho_s$</td>
<td>transmission loss of electricity at data center $s$</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>server acquisition cost</td>
</tr>
<tr>
<td>$\delta_s$</td>
<td>carbon tax at data center $s$</td>
</tr>
<tr>
<td>$M^{min}_s$</td>
<td>minimum number of servers at any data center</td>
</tr>
<tr>
<td>$M^{max}_s$</td>
<td>maximum number of servers at any data center</td>
</tr>
<tr>
<td>$v_{su}$</td>
<td>bandwidth cost for data center $s$ to data center $i$</td>
</tr>
<tr>
<td>$\xi_s$</td>
<td>number of bytes required for data replication of single request</td>
</tr>
</tbody>
</table>

### Decision Variables

<table>
<thead>
<tr>
<th>Decision Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m_s$</td>
</tr>
<tr>
<td>$\lambda^{fh}_{su}$</td>
</tr>
<tr>
<td>$y_{su}$</td>
</tr>
</tbody>
</table>

### Cost Components

| $I$              | total cost of ownership, including server acquisition cost, operating cost and data replication cost |
| $\Phi$           | server acquisition cost |
| $\eta$           | cost of data replication to nearest data center for durability |
| $\kappa$         | cost of data replication, for the case of replication at multiple sites |
| $\Theta$         | power consumption cost |
| $\tau$           | carbon tax incurred |

### Table II: Summary of notation used in the paper
availability.

**Delay:** Let \( D_{\text{max}} \) be the maximum latency for the service and \( D_{su} \) be the propagation delay between client region \( u \) and data center site \( s \). A data center must be assigned to the client region such that even after the failure of a site, the latency continues to be lower than \( D_{\text{max}} \).

**Server Utilization:** Let the processing rate of the server be \( s \) bps and let \( J_s \) be the response size for an application type \( a \in A \). The service rate for type \( a \) is defined by \( \frac{1}{J_s} \) requests per second. There are three approaches to model the average utilization of servers as given below:

1) **Mutually Exclusive (ME) approach:** Each type of application is assigned to a pre-defined set of servers. Let \( m_{sa} \) be the number of servers allocated to serve the requests of type \( a \). Requests for different services are queued in a single queue, from which a scheduler dispatches the requests to the corresponding servers. The average utilization of servers serving the requests of type \( a \) can be defined as

\[
\gamma_{s,a} = \frac{\sum J_a \lambda^a_{su}}{m_{sa} B},
\]

This approach of scheduling simplifies the resource provisioning but leads to under-utilization of servers.

2) **Maximum (MAX) approach:** Assuming all the requests to be homogeneous, the servers can be provisioned according to the highest processing rate required. In this case, the average utilization of any server can be defined as

\[
\gamma_{s} = \frac{\sum J_a \lambda_{su} \lambda^a_{su}}{m_{a} B},
\]

where \( J_{\text{max}} \) is the maximum mean file size across different application types. This approach also suffers from resource under-utilization. On the other hand, provisioning based on the smallest processing rate leads to under-provisioning of resources.

3) **Multiplexed (MUX) approach:** In a virtualized environment, any type of workload can be served by one of the free servers. All the requests are placed in a common queue and served by a set of identical servers. This model is followed in most of the recent papers [22]. The average utilization of a server in this case can be defined as

\[
\gamma_{s} = \sum J_a \lambda^a_{su} \lambda_{su},
\]

In this paper, we consider this model for server utilization but study the implications of other models in Section IV-B8.

**C. Cost Models**

**Server Acquisition:** Let the cost of a server normalized over the duration considered for evaluation be denoted by \( \alpha \). The total cost of servers across all the data centers, denoted by \( \Phi \) is simply given as

\[
\Phi = \alpha \sum m_{s}
\]

**Data Replication:** Let \( \nu_{sg} \) be the bandwidth cost for data replication from data centers \( s \) to \( g \). For every request served by a data center \( s \), let \( \xi \) be the volume of data to be replicated. We consider two possible replication models.

1) **Single site replication:** In this case, the data from a primary data center is replicated to the nearest data center.

2) **Multiple site replication:** In this case, the data from a primary data center is replicated to all possible data centers where the client’s request may be routed (denoted by \( PD_s \)), without exceeding the latency bound.

We define the cost of replication \( R \) for these two options using the following equation.

\[
R = \begin{cases} \sum_{a,f,u,s,h} \left( \lambda_{su}^a \xi \nu_{sg} \right) & \text{Case 1} \\ \sum_{a,f,u,s,h} \left( \lambda_{su}^a \xi \sum_{i \in P D_s} \nu_{si} \right) & \text{Case 2} \end{cases}
\]

**Power Consumption:** Let \( \omega_s \) denote the electricity price at data center location \( s \) in hour \( h \) of the day. Let \( P_{idle} \) be the average power consumed in idle condition and \( P_{peak} \) be the power consumed at peak utilization. Let \( E_s \) be the PUE of a data center. The total power consumed at \( s \) over an hour \( h \) can be expressed as [9]

\[
P_{s}^{h} = m_{s}\left(P_{idle} + \left(E_{s} - 1\right)P_{peak}\right) + m_{s}(P_{peak} - P_{idle})\gamma_{s}^{fh}.
\]

The total cost of power consumed at all the data center locations can be expressed as

\[
\Theta = \sum_{s,h} \omega_{h} P_{s}^{h}
\]

**Carbon tax:** Let \( \delta_{s} \) denote the carbon tax levied at data center location \( s \) and \( \rho_{a} \) denote the transmission loss incurred. The total cost due to carbon tax is

\[
\tau = \sum_{s,h} \delta_{s}(\rho_{s} + 1)P_{s}^{h}
\]

**D. CACP Model**

Considering all the cost factors defined above, we define the CACP problem as the problem of minimizing the TCO subject to the set of constraints on latency and availability. The TCO, denoted by \( F \), is defined as the sum of the server cost \( \Phi \), data replication cost \( R \), electricity cost \( \Theta \), and carbon tax \( \tau \). For notational simplicity, we define the following decision variables:

\[
m \triangleq \left\{ m_{s} \right\}, \forall s \in S,
\]

\[
\lambda \triangleq \left\{ \lambda^{a}_{su} \right\}, \forall s \in S, \forall u \in U, \forall a \in A, \forall h \in H, \forall f \in \{0,1,2,..,S\}
\]

and

\[
y \triangleq \left\{ y_{su} \right\}, \forall s \in S, \forall u \in U
\]

The CACP problem can be formally expressed as

\[
\text{minimize}_{m,\lambda,y} F = \Phi + R + \Theta + \tau
\]
subject to,

\[
\sum_{s \in S} \lambda_{su}^{afh} = L_u^{ah} \quad \forall u, a, h, f (13)
\]
\[
0 \leq \lambda_{su}^{afh} \leq y_{su} L_u^{ah} \quad \forall s, u, a, h, f (14)
\]
\[
M_{min} \leq M_s \leq M_{max} \quad \forall s (15)
\]
\[
P_{s}^{fh} \leq P_{s}^{h_{max}} \quad \forall s, f (16)
\]
\[
2D_{su} y_{su} \leq D_{max} \quad \forall s, u (17)
\]
\[
\gamma_s^{fh} \leq \gamma_{max}^{s} \quad \forall s, h, f (18)
\]
\[
\lambda_{su}^{afh} = 0 \quad \forall u, a, h, s = f (19)
\]

Among the constraints, Eq. 13 ensures that the demands of all client regions in every hour are met. Eq. 14 ensures that all the client requests are served by data centers within the latency limit. Eq. 15 ensures that capacity limit of a data center (in terms of number of servers) is not exceeded. The constraint on the total power available at a data center is taken care by Eq. 16. Eq. 17 ensures that the delay experienced by a client lies within the maximum bound. Eq. 18 is used to limit the queuing delay at a data center by bounding the average server utilization to a constant value ($\gamma_{max}^{s} \in (0, 1]$). Eq. 19 ensures that no demand is served by the failed data center.

The inputs to the CACP problem are: the set of data center locations with the associated costs, maximum average utilization of servers, processing rate of the servers, maximum latency, demand distribution, maximum number of servers at each site, and maximum power available at each site. The model then gives the number of servers across the sites, requests routing to the data centers, and the data centers within the latency limit for each client location.

E. Complexity Analysis

The number of variables in the above formulation is \(S + (S + 1)SU \times AH\) and the number of constraints is \(S + (S + 1)\{UAH + SUA \times AH + 2SH\} + SUA \times AH + SU\). The asymptotic complexity of proposed CACP model is \(O(S^2UAH)\). With an increase in the number of data centers, the complexity increases quadratically but linearly with the number of client locations, time slots, and application types. The following theorem states the complexity of the problem.

**Theorem 1.** The feasibility problem of CACP in a distributed data center is NP-hard.

**Proof.** The CACP problem in a distributed data center (without fault tolerance) is NP-hard, even when resources are of unit size and unit operating cost. The reduction is from the set cover problem. Detailed proof is given in the Appendix.

Though the formulation is NP-hard, solving it is a one-time effort only at the time of design. We do not see the running time to be a matter of concern since the CACP problem is always solved offline. Currently, the number of data centers hosted by data center operators is small (15 for Google [23]). We solved all the models centrally using CPLEX with Matlab on a server with an Intel Xeon processor, 64 GB of RAM, and 64-bit OS. We could not solve the model for more than ten data centers on this server in a reasonable amount of time (few minutes) for an evaluation period of one day (24 hourly slots).

We can solve the model optimally for capacity planning in large data centers with higher computational power. For much larger number of variables, we need to go for online heuristics or approximate algorithms.

F. Example for Working of the CACP Model

In this section, we give a simple example to illustrate the impact of the CACP model on the TCO. The proposed CACP model mainly reduces the TCO by exploiting demand multiplexing and spatio-temporal variation in the demand and electricity price. For easier understanding on how this works, we show two examples for (a) the impact of demand multiplexing on capacity provisioning and (b) the impact of demand multiplexing and electricity price variation on the TCO.

**Impact of demand multiplexing on capacity provisioning:**

Both the CACP and MS models take into account demand multiplexing while provisioning capacity, whereas the CDN model trivially maps requests to the nearest data center to minimize the latency. Consider a scenario with three data centers and three client regions with a maximum latency bound of 25 ms for the service. Fig. 1 shows the system used for illustration. Data centers DC1, DC2, and DC3 serve the requests from client regions C1, C2, and C3 given in Table III. Each edge between a data center and client region is weighted by the propagation delay. For simplicity, we consider the case where all the data centers are within the latency bound (25ms) for all the client regions.

| TABLE III: Demand from different client regions across three time slots (in units) |
|-----------------|-----------------|-----------------|
| Client 1        | Slot 1          | Slot 2          | Slot 3          |
| Client 2        | 100             | 50              | 50              |
| Client 3        | 50              | 50              | 100             |

![Fig. 1: System used for illustration](image)

Considering the case of a data center failure, a demand of 200 units generated from all the client regions needs to be served by the remaining two data centers. In the MS model, the workload is equally distributed among all the active data centers. This requires 100 servers at each data center and the total number of servers to tolerate any data center failure is
300 units as shown in Fig. 2b. In case of the CDN model, a client region is always served by the nearest data center after failure. For example, \( C_1 \) is served by \( DC_1 \) before failure, whereas it is served by \( DC_2 \) after failure. Therefore, \( DC_2 \) should be provisioned not only to satisfy \( C_2 \)’s demand but also with sufficient spare capacity to make up for the failed data center \( DC_1 \). This gives rise to \( DC_2 \) being provisioned with 150 servers to meet the demand across any interval (when \( DC_1 \) might fail). Accounting for the possibility of any data center failure, the server distribution across all the data centers is obtained to be 150 units as shown in Fig. 2a. We can conclude that the MS model exploits demand multiplexing while satisfying the latency bound of 25 ms and requires only 300 servers against 450 units with the CDN model. CACP model also gives the same result if we ignore the variation in the operating cost across the data centers.

**Impact of demand multiplexing and electricity price variation on the TCO:** Consider the three data centers shown in Fig.1 with the electricity price variation as shown in Fig. 3. It may be noted that the electricity price is highest at \( DC_2 \). The demand across the three regions \( C_1 \), \( C_2 \), and \( C_3 \) is shown in Fig. 4. We consider the processing rate to be 100/sec, \( P_{peak} \) and \( P_{idle} \) to be 400 W and 200 W, respectively, and the server cost to be $2000 (17 cents/hr, assuming 4 years life). We assumed that all the data centers are within the latency limit for any client region.

The server distribution obtained after solving the optimization model for CDN (minimize average latency), MS (minimize number of servers), CACP (minimize total cost) is shown in Fig. 5a, Fig. 5b, and Fig. 5c, respectively. The number of servers allocated across all the data center locations is the same with the MS model. However, the CACP model allocates fewer servers at \( DC_2 \), where the electricity price is higher. The CACP model always allocates more capacity at a site where the electricity price is cheaper while satisfying the latency and other constraints.

The normalized TCO obtained using the CACP model (with respect to the CDN model) is given in Table IV. Even though the CACP model allocates a larger number of servers than the MS model, the TCO is lowered by exploiting the spatio-temporal variation in the electricity prices for cost-aware demand distribution. Though the MS model minimizes the number of servers provisioned at each location, it does not give the minimum TCO because of being oblivious to the operating cost.

**IV. Numerical Results**

In this section, we solve the CACP model using real-world data and compare the TCO obtained with two other models from the literature. The MILP is solved using CPLEX.
CDN model: In this model, the objective is to balance the geographical distance in the order of 10 ms for every 1000 km [9].

MS model: A rudimentary version of this model was defined in [21]. The main objective of this model is to minimize the total number of servers deployed across all the data centers. The TCO for this model would be the cost of that data center provisioned after minimizing the number of servers.

CDN model: In this model, the objective is to balance the load across the data centers so that the average response time is minimized. The provisioning of servers in this model would be done so that the client latency is minimized [24].

We compared the TCO obtained using all three models in the results. We also studied the advantages of the CACP model by varying the number of data centers, demand, request rate, and latency bound. We also studied the impact of server utilization models and replication models discussed earlier on the TCO. We first provide details on the scenarios used and the data set used for the evaluation.

A. Scenarios Used

Data center locations: The locations for the data centers were (10 of them): California, Oregon, Virginia, Switzerland, U.K, Ireland, Netherlands, Hong Kong, Japan and Canada. At each location, the number of servers was varied between 1000 and 100,000. This would help us evaluate smaller and mega data centers across the world.

Client locations: Based on the data collected for the number of Internet users from [25] we selected the following client regions (15 of them): Brazil, China, Egypt, France, Germany, India, Indonesia, Japan, Mexico, Nigeria, Russia, South Korea, UK, USA, and Vietnam. The propagation delay between the data center location and client location varied linearly with geographical distance in the order of 10 ms for every 1000 km [9].

Electricity Prices: We used historical industrial electricity price data ($ per MWh) from publicly available government databases corresponding to various data center locations [11]–[14]. For the sake of brevity, we do not discuss regulated electricity market prices. Interested readers may see [11]. We use the electricity price model similar to the one in [18], where the price at each location is different during on-peak hours (7 A.M.-11 A.M. and 5 P.M.-7 P.M.), mid-peak hours (11 A.M-5 P.M.) and off-peak hours (7 P.M.-7 A.M.). The price varies across the periods by as much as 3 cents/kWh [18]. Some states in the USA (like California and Colorado) also add about $0.04 to $0.6/kWh as a carbon tax for power consumed from brown energy sources. Though our model includes carbon tax, we ignore the same in the results due to its small contribution in the TCO (less than 1%).

Traffic model: For the traffic we used the trace of requests to Wikipedia services downloaded from [15]. We downloaded the workload traces for the month of December 2015, containing the total number of requests and aggregate response size for different services of Wikipedia. The demand profile for a 24-hour period, averaged over a month, is plotted in Fig. 6. Since the demand had a diurnal pattern we used $H = 24$. This distribution of requests was used to derive an hourly demand for different client regions. We upscaled the number of requests by a factor of 3000 to reflect the traffic handled by larger service providers [17]. For each client region, we divided the workload proportional to the number of Internet users in that region. Table V shows the distribution of workload across different client regions obtained from the number of Internet users. Fig. 7 shows the hourly demand for a few client regions. The demand during the on-peak period was fixed at 1.4 times the mid-peak demand and the demand during the off-peak period was fixed at 0.6 times that in the mid-peak period.

Inter data center communication cost: For the inter-data center communication cost we used a pricing model similar to the one charged by AWS EC2 services [26]. For example, AWS charges $0.12 – $0.2/GB across geographical regions.

Other parameters: $P_{idle}$ and $P_{peak}$ were set to 200W and 400W, respectively [27]. The average PUE was set to 1.5 [28]. $P_{max}$ was taken as 100MW/hr for all the locations [18].

Table IV: Comparison of the normalized TCO for the models

<table>
<thead>
<tr>
<th>Models</th>
<th>Normalized TCO</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDN</td>
<td>1</td>
</tr>
<tr>
<td>MS</td>
<td>0.89</td>
</tr>
<tr>
<td>CACP</td>
<td>0.62</td>
</tr>
</tbody>
</table>

Fig. 5: Capacity allocation using (a) CDN model, (b) MS model, (c) CACP model
1.8 days of failure per year.

B. Results

In this section, we present the results from evaluating the models for the TCO by varying the number of data centers, demand, and latency bound. We also study the effect of different models for server utilization and data replication (single site and multi-site) on the TCO. In all the results, we show the normalized values of TCO, where the normalization was done using the maximum TCO in all the experiments.

1) TCO comparison:
In this experiment, we varied the number of data centers between 6 and 10 in order to serve the client requests as reported earlier within a maximum latency of 300 ms. Fig. 8 shows the normalized TCO for all the models with various numbers of data centers. In this experiment, we used the single site replication model.

Table VI reports the normalized TCO for different cases (third row from the bottom). Reduction in the TCO (percentage) with the CACP model (compared to the MS and CDN models) is shown in the last two rows. The fourth row from the bottom shows the total number of servers provisioned with each model across the data centers. The table also shows the locations chosen and the number of servers at each location as the desired number of locations increased. Since the CACP model exploits the spatio-temporal variation in the electricity prices, the TCO is lowest in the case of the CACP model.

Though the MS model minimizes the number of servers provisioned at each location, it does not lead to a minimum TCO. This is because it does not consider the operating cost. From Table VI it can be observed that even with six data centers, the benefit of the CACP model is significant, while the other two models have a similar TCO. This is due to the fact that with fewer data centers, there is not much scope for demand multiplexing. On the other hand, the CACP model assigns a larger workload at a data center location with a lower electricity price. With addition of another location (the Netherlands, with a lower electricity price compared to the U.K. and Ireland), the CACP model shifts the servers provisioned in the U.K. and Ireland to the Netherlands (see Table VI). This improves the TCO in the CACP model by about 3.5%. While the CACP model suggests more servers, the TCO is minimized due to shifting them to locations with a lower operating cost. This can be observed from the table that shows the MS model gives the same number of servers at each location. We can observe that the CACP model achieved a TCO reduction of up to 35% compared to the MS model, and up to 43% compared to the CDN model.

2) Impact of data center locations on the TCO:
We also studied how the choice of data center locations affects the TCO with the CACP model. We evaluated our model for the following sets of locations:
Both these factors lead to a lower TCO for the USA) also meets the latency constraints for the largest number of users (from Americas as reported in Table V).

Next, we studied the impact of maximum latency bound (w.r.t CDN) also meets the latency constraints for the largest number of users (from Americas as reported in Table V). Both these factors lead to a lower TCO for Set 2 than Set 5.

3) Impact of Replication Cost:

To understand the contribution of the replication cost to the TCO, we evaluated the CACP model considering the single site replication (SR) and multiple site replication (MR) models with various numbers of data centers. The maximum latency was set to 300ms and the demand was generated as reported in Section IV-A. Fig. 10 shows the TCO split into the replication cost and the cost due to power consumption for both replication models. It can be observed that in the SR model, the contribution of the replication cost is low in the TCO. On the other hand, the MR model is costly for replication and the replication cost increases with the number of data centers as shown in Fig 10. Therefore, this approach may be preferred only when the post-failure penalty is very high.

Fig. 11 shows the TCO for the CACP model with and without the replication cost being considered. It can be observed that the single site replication cost alone accounts for 20% of the TCO. Therefore, the CACP model (without replication cost) lowers the TCO by about 20% compared to the model with replication.

In all the subsequent experiments, we considered only the single site replication model while evaluating the TCO.

4) TCO vs. Worst-case Latency:

Next, we studied the impact of maximum latency bound
on the TCO. We evaluated the models for 8 data centers, 15 client regions, and the aggregate demand as mentioned in Section IV-A. The maximum latency was chosen in the range of 150 – 350 ms. Fig. 12 shows the normalized TCO for all the models with varying latency. We can observe that the CACP model results in a lower TCO by upto 38% and 32%, respectively compared to the other models. In the CACP model, there is a choice in the number of data centers capable of serving the requests from a particular client region that leads to better multiplexing of resources and a reduced TCO. Apart from this, the CACP model also selects the data centers in regions with lower electricity prices while meeting the latency bound. Although the CDN model gives minimum latency, request routing is oblivious to the variation in the electricity price. Therefore, the TCO is higher with the CDN model particularly when the latency requirements are not very stringent. We conclude that the CACP model is more advantageous for services without stringent latency requirement.

We also show how the worst-case latency increases when the CACP model targets the TCO reduction (as compared to the CDN model) in Table VII. At a worst-case latency of 150 ms, our model has about a 25% lower TCO. When we target a higher reduction in the TCO, the worst-case latency in the CACP model increases. For about 40% reduction in the TCO, our model leads to a worst-case latency of 300 ms. The reduction in the TCO is achieved because the CACP model exploits demand multiplexing and variation in the electricity price when the latency requirement is relaxed.

6) Impact of demand multiplexing:
To study the impact of demand multiplexing on the TCO, we evaluated the models by varying the number of data centers from 6 to 10. The electricity price was fixed at 10 cents per kWh throughout the day and the replication cost was fixed to $0.2/GB. The delay bound was set to 300ms. It can be observed from Fig.14 that CACP and MS models give a lower TCO than the CDN model, since CDN model does not allow demand multiplexing due to the latency minimization objective. The CACP model reduces the TCO by almost 45% compared to the CDN model. We also noticed that the CACP model eventually gives the same TCO as the MS model, because cost reduction is only possible by demand multiplexing that minimizes the total number of servers (due to a uniform electricity price). The TCO reduction of about 10% can be attributed to demand multiplexing as the number of data centers increases.

5) Impact of Demand:
We evaluated all the models by varying the total demand for 8 data centers and a maximum latency bound of 300ms. Results in Fig. 13 show that as the demand increases, the TCO for the CACP model is lower compared to other models. Due to the capacity limit of a data center, increased demand causes saturation of all the data centers in the regions with lower electricity price. This reduces the choices available and leads to the selection of other locations with higher electricity prices. The proposed model is advantageous only when the data center does not operate at peak utilization. Under a heavy load, the CACP model can help the provider determine an optimal data center upgrade plan while minimizing the TCO.

7) Cost of over provisioning:
To study the cost of over-provisioning for fault tolerance, we evaluated all the models by varying the number of data
In this paper, we addressed the problem of cost-aware capacity provisioning for geo-distributed data centers capable of masking single data center failure. We prove that this problem is NP-hard and proposed an MILP formulation to reduce the TCO. The proposed model is observed to be better than the MS and CDN models due to its ability to multiplex demand considering the spatio-temporal variation in the electricity prices and the demand. We also modeled different approaches to serve heterogeneous demand and data replication. Numerical results demonstrated that the approach of minimizing the TCO is beneficial when the electricity price varies significantly, which appears to be the case for most of the cloud providers operating geo-distributed data centers. The CACP model achieves a cost reduction of up to 34% and 50% when compared to the MS and CDN models, respectively. Our model is also useful to study the effect of the replication cost on the TCO for planning distributed data centers.

V. CONCLUSION

In this paper, we addressed the problem of cost-aware capacity provisioning for geo-distributed data centers capable of masking single data center failure. We prove that this problem is NP-hard and proposed an MILP formulation to reduce the TCO. The proposed model is observed to be better than the MS and CDN models due to its ability to multiplex demand considering the spatio-temporal variation in the electricity prices and the demand. We also modeled different approaches to serve heterogeneous demand and data replication. Numerical results demonstrated that the approach of minimizing the TCO is beneficial when the electricity price varies significantly, which appears to be the case for most of the cloud providers operating geo-distributed data centers. The CACP model achieves a cost reduction of up to 34% and 50% when compared to the MS and CDN models, respectively. Our model is also useful to study the effect of the replication cost on the TCO for planning distributed data centers.

APPENDIX

In this section, we prove Theorem 1 stated in the paper. In a basic formulation, the cost aware capacity provisioning problem (without failure considerations) consists of a set of data center locations DC where the cost of running servers at a data center i is given by Costi, and a set of client locations C generating a demand to be served. Each client can be served by a data center lying within a given latency bound Delay. The goal is to provision a number of servers across data centers so that the total cost incurred is minimum while satisfying client demand and latency bound.

In Lemma 1 we reduce the decision version of the set cover problem to the decision version of the CACP problem, which is sufficient to show that Theorem 1 holds. Formally, the decision version of the CACP problem is defined as follows. Given a set of data centers and their server running costs, a set of demand generating client regions and latency bound, does there exist a subset of data centers that can satisfy the client demand with the total cost incurred being at most k?

**Lemma 1.** The decision version of the CACP problem is NP-hard.
Proof. The decision version of the set cover problem is defined as follows. Given a set system \((\mathcal{U}, \mathcal{S})\) with \(\bigcup_{S \in \mathcal{S}} S = \mathcal{U}\) and a positive integer \(k\). The question is does there exist a collection of \(k\) or fewer sets of \(\mathcal{S}\) that cover \(\mathcal{U}\) [31]? This problem is known to be NP-complete and we give a reduction of this problem to the decision version of the CACP problem as follows.

Given an instance of the set cover problem \(I_S\), let us map it to an instance \(I_C\) of the decision version of the CACP problem. For each \(u \in \mathcal{U}\), we assign a client region \(c_u\) that generates a demand of unit compute capacity to meet its needs. For each \(S \in \mathcal{S}\), we assign a data center \(d_S\) that is within the delay bound for the clients specified as its elements. For instance, if \(S = \{u_1, u_2, \ldots, u_m\}\) then \(d_S\) has \(c_{u_1}, c_{u_2}, \ldots, c_{u_m}\) within the delay bound constraint. The cost associated with each data center is 1 unit and each of them has infinite capacity. This completes the reduction of instance \(I_S\) to \(I_C\). It is easy to observe that the reduction from \(I_S\) to \(I_C\) is in polynomial time in the input size of instance \(I_S\). To complete the proof, we need to show that \(I_S\) admits a solution if and only if \(I_C\) has a solution that costs at most, \(k\) units.

Suppose \(I_C\) has a solution with less than or equal to cost \(k\) units. Without loss of generality, let \(d_{S_1}, d_{S_2}, \ldots, d_{S_l}\) be the solution to \(I_C\) that meets demands of all client regions. Note that \(l \leq k\) as each data center consumes 1 unit of energy. Each of the clients \(c_u\) is served by at least one data center in \(d_{S_1}, d_{S_2}, \ldots, d_{S_l}\). Correspondingly, the \(S_1, S_2, \ldots, S_l\) cover each \(u \in \mathcal{U}\) and thus, it is a solution to \(I_S\) having the size of \(l \leq k\).

Conversely, if \(I_S\) admits a solution \(S_1, S_2, \ldots, S_j\) with \(j \leq k\) we can construct a solution to \(I_C\) that costs at most, \(k\) units. The set of data centers \(d_{S_1}, d_{S_2}, \ldots, d_{S_l}\) is able to meet the demand of all the client regions \(c_u\) as \(\bigcup_{1 \leq i \leq j} S_i = \mathcal{U}\). Thus we have constructed a solution to \(I_C\) that costs \(j \leq k\) units.

\[\square\]

REFERENCES


S Vignesh received his B.Tech. degree from NIT, Tiruchirappalli, India, and the M.Tech. degree from the Indian Institute of Technology Guwahati, India. His research interests include data center networks, distributed systems, and Internet of Things.

Venkatesh Tamarapalli is an Associate Professor with the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati. He received Ph.D from Indian Institute of Technology Madras in 2009. He was a recipient of Microsoft Ph.D. Fellowship and Microsoft Young Outstanding Faculty Award. He co-authored a book titled *An Analytical Approach to Optical Burst Switched Networks* published by Springer USA. He is currently serving as an Associate Editor for Springer Journal of Network and Systems Management. His areas of research include wireless sensor networks, data center networks, and multimedia content delivery.

Deep Medhi is Curators’ Distinguished Professor in the Department of Computer Science & Electrical Engineering at the University of Missouri-Kansas City, USA, and a honorary professor in the Department of Computer Science & Engineering at the Indian Institute of Technology–Guwahati, India. He received B.Sc. in Mathematics from Cotton College, Guahati University, India, M.Sc. in Mathematics from the University of Delhi, India, and his Ph.D. in Computer Sciences from the University of Wisconsin-Madison, USA. Prior to joining UMKC in 1989, he was a member of the technical staff at AT&T Bell Laboratories. He served as an invited visiting professor at the Technical University of Denmark, a visiting research fellow at Lund Institute of Technology, Sweden, and State University of Campinas, Brazil. As a Fulbright Senior Specialist, he was a visitor at Bilkent University, Turkey, and Kurukshetra University, India. He is the Editor-in-Chief of Springer’s *Journal of Network and Systems Management*, and serves (or served) on the editorial board of *IEEE/ACM Transactions on Networking, IEEE Transactions on Network and Service Management*, and *IEEE Communications Surveys & Tutorials, Telecommunications Systems, Computer Networks, and IEEE Communications Magazine*. He has published over 150 papers, and is co-author of the books, *Routing, Flow, and Capacity Design in Communication and Computer Networks* (2004) and *Network Routing: Algorithms, Protocols, and Architectures* (2007), both published by Morgan Kaufmann Publishers, an imprint of Elsevier Science. His research interests are: multi-layer networking; network virtualization; data center optimization; network routing, design, and survivability; and video quality-of-experience. His research has been funded by NSF and DARPA.