IEICE TRANS. COMMUN., VOL.E97–B, NO.1 JANUARY 2014

INVITED PAPER *Special Section on Management for Flexible ICT Systems and Services*

On Achieving High Survivability in Virtualized Data Centers

Md Golam RABBANI[†a)], **Mohamed Faten ZHANI**[†b)], *and* **Raouf BOUTABA**[†c)], *Nonmembers*

SUMMARY As businesses are increasingly relying on the cloud to host their services, cloud providers are striving to offer guaranteed and highly-available resources. To achieve this goal, recent proposals have advocated to offer both computing and networking resources in the form of Virtual Data Centers (VDCs). Subsequently, several attempts have been made to improve the availability of VDCs through reliability-aware resource allocation schemes and redundancy provisioning techniques. However, the research to date has not considered the heterogeneity of the underlying physical components. Specifically, it does not consider recent findings showing that failure rates and availability of data center equipments can vary significantly depending on various parameters including their types and ages. To address this limitation, in this paper we propose a High-availability Virtual Infrastructure management framework (Hi-VI) that takes into account the heterogeneity of cloud data center equipments to dynamically provision backup resources in order to ensure required VDC availability. Specifically, we propose a technique to compute the availability of a VDC that considers both (1) the heterogeneity of data center networking and computing equipments in terms of failure rates and availability, and (2) the number of redundant virtual nodes and links provisioned as backups. We then leverage this technique to propose an allocation scheme that jointly provisions resources for VDCs and backups of virtual components with the goal of achieving the required VDC availability while minimizing energy costs. Through simulations, we demonstrate the effectiveness of our framework compared to heterogeneity-oblivious solutions.

key words: *cloud computing, virtualization, data center management, survivability*

1. Introduction

In recent years, Cloud Computing has become the top platform for service hosting and delivery. In typical cloud environments, the Cloud Provider (CP) owns the physical infrastructure and leases resources to multiple Service Providers (SPs). Each SP then leverages its dedicated resources to deploy applications and services and offer them to end users over the Internet. So far, CPs mainly roll out computing resources (i.e., virtual machines) with no network performance guarantees [1], [2]. This results in a variable and unpredictable network performance in addition to potential security risks [3]–[6].

To address these issues, recent research proposals have advocated offering both computing and networking resources in the form of Virtual Data Centers (VDCs). Basically, a VDC consists of virtual machines, routers and switches connected through virtual links with guaranteed bandwidth. This allows CPs to achieve better performance isolation between VDCs and to implement more fine-grained resource allocation schemes. At the same time, VDCs allow SPs to guarantee predictable network performance for their applications.

A growing body of work has recently studied the problem of resource allocation for VDCs aiming to achieve several objectives such as minimizing energy costs and maximizing revenues [3], [4], [6]. However, one primary challenge that has not been adequately addressed so far is how to guarantee high VDC availability. On the one hand, for a SP, a service disruption, even for seconds, may incur high losses in revenue and also significantly impair the SP reputation. A recent study by an IT analyst firm estimated SP losses due to service downtime from $25,000 up to $150,000 per hour [7]. On the other hand, CPs could also incur significant penalties from not delivering the promised availability specified in the service level agreements. For instance, Amazon EC2 pledges to offer a service credit equal to 10% of the bill to any customer whose resources' annual availability falls below 99.95% [1]. As a result, providing high resource availability has been pointed out as one of the primary challenges of CPs to exhort SPs to rely on the cloud.

Recently, few proposals have been made to improve the availability of VDCs either through reliability-aware resource allocation schemes or redundancy provisioning techniques [8]–[12]. However, these works did not consider the heterogeneity of the underlying physical components. Indeed, it has been reported that failure rates and availabilities of the underlying physical components vary significantly depending on various parameters such as the type of equipments and their age [9], [12]. This observation suggests that in order to achieve the required VDC availability, the heterogenous availability of the infrastructure equipments must be considered to (1) determine the placement of the VDC components and to (2) estimate more accurately the number and the placement of redundant virtual nodes and links.

In this paper, we address this compelling challenge and propose **Hi**gh-availability **V**irtual **I**nfrastructure Management (Hi-VI) framework that takes into account the heterogeneity of data center computing and networking equipments to dynamically provision backup resources in order to ensure the required VDC availability. We first propose a technique to compute the availability of a VDC that considers both (1) the heterogeneity of data center networking and computing equipments in terms of failure rate and availabil-

Manuscript received June 10, 2013.
Manuscript revised August 23, 2013.

[†]The authors are with the D.R. Cheriton School of Computer Science, University of Waterloo, Canada.
a) E-mail: m6rabban@uwaterloo.ca
b) E-mail: mfzhani@uwaterloo.ca
c) E-mail: rboutaba@uwaterloo.ca

DOI: 10.1587/transcom.E97.B.10

ity, and (2) the number of redundant virtual nodes and links provisioned as backups. We then leverage this technique to propose an allocation scheme that jointly provisions VDC computing and networking resources as well as backups for virtual machines and links with the goal of achieving the availability required for each VDC while minimizing the resources used for backups and the total operational costs (e.g., energy costs) of the CP.

The remainder of the paper is organized as follows: Sect. 2 surveys previous work on failure characterization in data centers highlighting the heterogeneity in terms of failure rate and availability. We also summarize representative work on reliability-aware VDC allocation schemes and discuss their limitations. We present in Sect. 3 our technique to compute the availability of a VDC and we provide the mathematical formulation of the availability-aware VDC allocation problem. The proposed solution is then described in Sect. 4. Simulation results are discussed in Sect. 5. Finally, we draw our conclusions in Sect. 6.

2. Literature Survey

In this section, we present related work on data center failure characterization and representative proposals addressing survivable resource allocation in virtualized data centers.

2.1 Failure Characterization in Data Centers

Gill et al. [9] presented a large-scale study of failures in several Microsoft data centers over one year. The authors characterized failures of networking devices and assessed the impact of their failures and the effectiveness of network redundancy in data centers. Furthermore, they observed that the failure rates of different equipments can vary significantly depending on their type (servers, top-of-rack switches, aggregation switches, routers, load balancers) and model. For instance, Load Balancers (LBs) exhibit high probability of failure during one-year period (over 20%), whereas switches have lower failure probability (less than 5%). Furthermore, failure rates of different devices are unevenly distributed. For example, the number of failures across load balancers are highly variable with a few outlier LB devices experiencing more than 400 failures over the one year period. Finally, the analysis of failure traces revealed that correlated device and link failures are extremely rare.

Wu et al. [8] presented an automated failure mitigation system called *NetPilot*, which alleviates failures in large scale data center network before operators diagnose and repair the root cause. The authors built their system based on an analysis of failures in several production data center networks over a six-month period. They identified three main causes of failures: software failures which constitute 21% of the total number of failures, hardware failures accounting for 18% and finally misconfigurations, the most dominant source of the failures (38%). The authors found that usually simple steps of mitigation are very effective in reducing repair times. However, certain failures incur much more repair time and hence cause significant network downtime. This concurs with the finding of [13] that reported that more than 95% of network failures can be fixed within 10 minutes whereas at least 0.09% of them can take more than 10 days to resolve. This again shows the heterogenity of the failures in terms of repair times and potential impact.

Vishwanath et al. [14] analyzed failures of more than 100,000 servers deployed in multiple Microsoft data centers over a duration of 14 months. They found that hard disk, memory and raid controller failures were the main reason for server failures. For instance, they reported that failures of hard disks represent 78% of the total failures causing service disruption. They also noticed a high correlation between the number of disk drives in the server and the number of server failures. In addition, they found that servers that have experienced failures are likely to fail again in the near future. This results in a skewed distribution of server failure rate.

Based on these observations, we can summarize the main characteristics of failures in data centers as follows: (1) failure rates and availability are heterogenous across the physical components, (2) correlated failures are extremely rare. This suggests that heterogeneity should be considered when mapping virtual components onto the physical infrastructure. Furthermore, since correlated failures are rare, it is reasonable to assume failures to be independent.

2.2 Survivable VDC Embedding

Bodik et al. [11] proposed an allocation scheme that aims at minimizing the impact of failures (i.e., maximizing fault-tolerance) on the virtual data center (termed "service" in the paper) while reducing bandwidth usage in the core of the data center network. The VDC fault-tolerance is measured by the *worst-case survival* metric defined as the fraction of VMs belonging to the same VDC that remain operational during a single worst-case failure. However, this work does not consider the availability of the underlying physical components. Besides, considering only worst-case failure (which happens in aggregation/core switches) results in ignoring other failures (e.g., in top-of-rack switches). Furthermore, the authors assume a physical server can only host one VM from the same VDC. As a result, the approach tends to extensively spread VMs, leading to higher bandwidth usage.

Xu et al. [10] proposed a VDC allocation scheme that considers embedding backup VMs and virtual links with the goal of minimizing consumed resources. However, they do not consider the availability of the physical machines and they also assume that the number of backups is known beforehand. Yeow et al. [16] addressed these limitations and they proposed a reliable VDC embedding scheme that achieves the desired availability for VDCs by estimating the required number of backups for the virtual nodes based on the availability of physical machines. They also introduced a technique to allow VDCs to share backup nodes and links. However, this work considers only homogenous clusters, which means all servers have same probability of failure and availability, which is an unrealistic assumption.

Table 1 Comparison of survivable embedding schemes.

Proposals	Backup provisioning		Estimation of the number of backups	Heterogenity	Computing Availability	VM Colocation
	Virtual Nodes	Virtual Links				
Xu et al. [10]	Yes	No	No	No	No	Yes
Yu et al. [15]	Yes	No	No	No	No	No
Yeow et al. [16]	Yes	No	Yes	No	Yes	No
Rahman et al. [17]	No	Yes	N/A	N/A	No	N/A
Bodik et al. [11]	No	No	No	No	No	No
Hi-VI	Yes	Yes	Yes	Yes	Yes	Yes

They also assumed that a physical node cannot host more than one virtual node from the same VDC. Yu et al. [15] proposed a backup provisioning scheme for improving virtual infrastructure survivability while minimizing resources used to provision backups. Assuming that only a single failure could occur at a time, they formulate a Mixed Integer Linear Program (MILP) that estimates the required number of redundant nodes and their placement in order to minimize networking resources provisioned for the backup nodes.

Rahman et al. [17] presented two policies for solving survivable virtual network embedding problem. The first policy addresses failures proactively by provisioning backup paths for potential failures in the future, however, this approach may lead to the wastage of up to 50% of physical resources. The second policy heuristic is a reactive approach that precomputes a set of possible backup detours for each substrate link. When a substrate link fails, the affected virtual links are rerouted along one of the backup detours. However, this approach does not consider multiple link failures.

Table 1 compares the features of survivable embedding proposals. The limitations of the state of the art research can be summarized as follows:

- Previous proposals have either ignored the availability of the underlying physical components (e.g., [10], [11]) or considered that the cluster is homogenous, i.e., nodes have similar failure rates and availability (e.g., [16]). Hence, it is more realistic and challenging to consider the heterogeneity existing in production data center environments in order to take more informed resource allocation decisions and improve availability of the embedded VDCs.
- Existing proposals (e.g., [15], [16]) assume a single physical server can host at most one virtual node from the same VDC. This assumption is not realistic in production environments. For instance, if a VDC comprises hundreds of VMs, these schemes map them onto hundreds of physical servers. This results in higher bandwidth consumption and requires more physical nodes to be active. Ideally, it should be possible to allow multiple VMs from the same VDC to be hosted on a single physical node if the required availability is satisfied. This will result in reduced bandwidth usage and less active physical nodes, and lead to reduced enegy costs, increased VDC acceptance and CP revenue.
- Previous work (e.g., [16]) does not consider the failure rate of networking components (e.g., physical switches) when computing the VDC availability. However, virtual links are mapped onto physical paths that may cross multiple physical switches. It is therefore mandatory to factor in switches' availability when computing the availability of VDCs.

In this paper, we aim to address these limitations by developping a technique to estimate VDC availability in a heterogeneous environment and then leverage it to devise a more efficient resource allocation scheme that achieves availability requirements and at the same time minimizes energy costs.

3. Survivability in Virtualized Data Centers

In this section, we provide a technique to compute the VDC availability and a mathematical formulation of the availability-aware embedding problem.

3.1 VDC Availability in Heterogenous Environments

Once the SP provides the requirements of his VDC in terms of resources and availability, the CP is responsible for mapping the VDC onto the physical data center such that the required availability is satisfied. Hence, the CP should be able to (1) evaluate the availability of the embedded virtual components based on the availability of the underlying physical infrastrucure, and to (2) estimate the number of backups needed to meet the required availability.

In the following, we first describe how we model the physical data center and the VDC requests. We then present a technique to compute the availability of a VDC taking into consideration the heterogenous characteristics of the physical equipments. Finally, we formulate the survivable VDC embedding problem as an optimization problem that minimizes the number of active physical machines, the bandwidth usage in the data center network as well as the number of backups while satisfying the required VDC availability.

3.1.1 Physical Data Center

We model the data center network as a graph $\bar{G} = (\bar{N}, \bar{L})$ where $\bar{N}$ is the set of physical nodes and $\bar{L}$ is the set of physical links. $\bar{N}$ includes the set of physical machines $\bar{M}$ and the set of physical switches and routers $\bar{S}$ (i.e., $\bar{N} = \bar{M} \cup \bar{S}$).

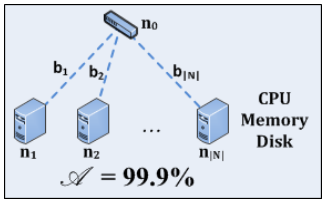
Each physical node $\bar{n} \in \bar{N}$ has a residual capacity $c^r_{\bar{n}}$ for each resource type $r \in R$ where $R = \{1 \dots |R|\}$ is the set of resource types. Each link $\bar{l} \in \bar{L}$ has a residual bandwidth capacity $b_{\bar{l}}$. The availability $\mathscr{A}_{\bar{n}} \in [0, 1]$ of a physical component $\bar{n}$ is given by:

$$\mathscr{A}_{\bar{n}} = \frac{MTBF_{\bar{n}}}{MTBF_{\bar{n}} + MTTR_{\bar{n}}} \tag{1}$$

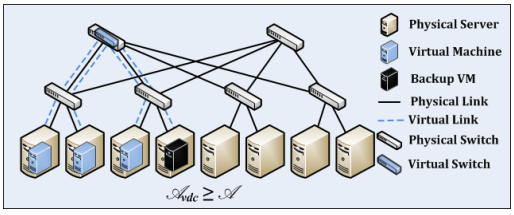
where $MTBF_{\bar{n}}$ and $MTTR_{\bar{n}}$ represent respectively the Mean Time Between Failures and the Mean Time To Repair for the node $\bar{n}$. Both $MTBF_{\bar{n}}$ and $MTTR_{\bar{n}}$ can be obtained from historical records of failure events. Furthermore, we define $\bar{u}_{\bar{n}\bar{l}}$ and $\bar{v}_{\bar{n}\bar{l}}$ as boolean variables that indicate whether a physical node $\bar{n} \in \bar{N}$ is the source and the destination of physical link $\bar{l} \in \bar{L}$, respectively.

3.1.2 VDC Requests

In this work, we limit our study to VDC requests having a star topology as shown in Fig. 1. Such a virtual topology is suitable for hosting many types of applications like web applications, MapReduce and BLAST [3]. Hence, a SP has to specify the number of virtual nodes constituting the VDC, the amount of resources for each of the VMs (i.e., CPU, memory and disk) and links (i.e., bandwidth) as well as the required VDC availability (see Fig. 1(a)). Similar to a physical data center, a VDC request can be modelled as a graph $G = (N, L)$, where N is the set of virtual nodes (including the virtual switch) and L is the set of virtual links. The required availability of the VDC is denoted by $\mathscr{A}$. Each virtual node $n \in N$ has a capacity c^r_n for each resource type $r \in R$, and each virtual link $l \in L$ has a bandwidth capacity b_l. Since we have only a single virtual switch, we reserve for it the index 0. We also define two boolean variables u_{nl} and v_{nl} to indicate whether a virtual node $n \in N$ is the source or the destination of virtual link $l \in L$, respectively.



(a) Format of the VDC request.



(b) VDC Mapping (with 1 backup node).

Fig. 1 A sample VDC request and its embedding in physical data center.

3.1.3 Variable Definitions

We hereafter define variables that capture the mapping of virtual nodes and links onto the physical infrastructure. Let $x_{n\bar{n}} \in \{0, 1\}$ be a boolean variable that indicates whether virtual node n is mapped onto the substrate machine $\bar{n}$. Let $f_{l\bar{l}} \in \{0, 1\}$ be a boolean variable that indicates whether physical link $\bar{l}$ is used to embed virtual link l.

We also define $w_{n\bar{s}} \in \{0, 1\}$ that indicates whether physical switch $\bar{s}$ is used to embed the virtual link connecting the virtual switch to the virtual node n. In other words, if $w_{n\bar{s}} = 1$ the failure of physical switch $\bar{s}$ causes the virtual node n to be unavailable. Hence, $w_{n\bar{s}}$ can be expressed as:

$$w_{n\bar{s}} = \begin{cases} 1 & \text{if } \sum_{l \in L} \sum_{\bar{l} \in \bar{L}} (u_{nl} f_{l\bar{l}} u_{s\bar{l}} + u_{nl} f_{l\bar{l}} v_{s\bar{l}} \\ & + v_{nl} f_{l\bar{l}} u_{s\bar{l}} + v_{nl} f_{l\bar{l}} v_{s\bar{l}}) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $u_{nl} f_{l\bar{l}} u_{\bar{s}\bar{l}}$ indicates whether physical link $\bar{l}$ is used to embed virtual link l, and virtual node n is source of l, and physical node $\bar{s}$ is source of $\bar{l}$. We also define $y_{\bar{n}}$ as a boolean variable that indicates whether a physical node $\bar{n} \in \bar{N}$ is used either to embed a VM or switch or to embed a virtual link (as an intermediate node in the physical path).

3.1.4 Computing VDC Availability

In the following, we provide a technique to compute the availability of a VDC request $G = (N, L)$. Let N_B and L_B denote the set of backup nodes and links that are provisioned by the CP in order to improve the availability of the VDC. The resulting graph including the backup links and nodes is denoted by $G' = (N', L')$ where $N' = N \cup N_B$ and $L' = L \cup L_B$ where N_B and L_B are the set of backup nodes and links, respectively. Note that for the considered star topology we have $|N_B| = |L_B|$. A VDC is available if the number of failed virtual nodes is at most the number of provisioned backups. Let $Pr(k)$ be the probability that k virtual nodes fail. Hence, the availability of the whole VDC $\mathscr{A}_{vdc}$ can be written as:

$$\mathscr{A}_{vdc} = \sum_{k=0}^{K} Pr(k) = Pr(0) + \sum_{k=1}^{K} Pr(k) \tag{3}$$

Let us first compute the probability that no virtual node fails $Pr(0)$. It can be written as the product of the availability of all physical nodes hosting the VDC components:

$$Pr(0) = \prod_{\bar{n}: y_{\bar{n}} = 1} y_{\bar{n}} \mathscr{A}_{\bar{n}} \tag{4}$$

Next, let us compute the probability $Pr(k)$ where $k \geq 1$. The failure of k virtual nodes occurs only when physical failures result in k VM failures. Let $g_{\bar{m}}$ be the number of VMs mapped to physical machine $\bar{m}$. It can be written as:

$$g_{\bar{m}} = \sum_{n \in N'} x_{n\bar{m}} \quad \forall \bar{m} \in \bar{M} \tag{5}$$

Let $g_{\bar{s}}$ be the number of VMs that are disconnected if the physical swtich $\bar{s}$ fails. In other words, this switch is used either to embed the virtual switch or as an intermediate node between the physical server hosting the VM and the physical node hosting the virtual switch. We have:

$$g_{\bar{s}} = \sum_{n \in N'} w_{n\bar{s}} \quad \forall \bar{s} \in \bar{S} \tag{6}$$

To evaluate the probability of k virtual nodes failure, we need to consider every possible physical node failure that will lead to k virtual nodes failure. The probability of having a single physical node failure that causes k virtual nodes failure can be written as:

$$\sum_{\bar{n}: g_{\bar{n}}=k} \left((1 - \mathscr{A}_{\bar{n}}) \prod_{\bar{t} \in \bar{N} \setminus \{\bar{n}\}: y_{\bar{t}}=1} y_{\bar{t}} \mathscr{A}_{\bar{t}} \right)$$

where $(1-\mathscr{A}_{\bar{n}})$ is the probability of failure of physical node $\bar{n}$ and $\prod_{\bar{t} \in \bar{N} \setminus \{\bar{n}\}: y_{\bar{t}}=1} y_{\bar{t}} \mathscr{A}_{\bar{t}}$ is the probability that all other nodes used to embed the VDC are available. Note that we consider the failure of physical nodes that can impact k virtual machines (i.e., $g_{\bar{n}} = k$). However, in practice, multiple physical nodes can fail simultaneously and lead to k virtual node failure. Therefore, we have:

$$Pr(k) \geq \sum_{\bar{n}: g_{\bar{n}}=k} \left((1 - \mathscr{A}_{\bar{n}}) \prod_{\bar{t} \in \bar{N} \setminus \{\bar{n}\}: y_{\bar{t}}=1} y_{\bar{t}} \mathscr{A}_{\bar{t}} \right) \tag{7}$$

Using Eqs. (3), (4), and (7), we have:

$$\mathscr{A}_{vdc} \geq \left(\prod_{\bar{n}: y_{\bar{n}}=1} y_{\bar{n}} \mathscr{A}_{\bar{n}} \right) + \sum_{k=1}^{K} \left(\sum_{\bar{n}: g_{\bar{n}}=k} ((1 - \mathscr{A}_{\bar{n}}) \prod_{\bar{t} \in \bar{N} \setminus \{\bar{n}\}: y_{\bar{t}}=1} y_{\bar{t}} \mathscr{A}_{\bar{t}}) \right)$$

This provides a lower bound on the availability of the VDC. Let $\mathscr{A}^{lb}_{vdc}$ denote this lower bound, it can be written as:

$$\mathscr{A}^{lb}_{vdc} = \left(\prod_{\bar{n}: y_{\bar{n}}=1} y_{\bar{n}} \mathscr{A}_{\bar{n}} \right) + \sum_{k=1}^{K} \left(\sum_{\bar{n}: g_{\bar{n}}=k} ((1 - \mathscr{A}_{\bar{n}}) \prod_{\bar{t} \in \bar{N} \setminus \{\bar{n}\}: y_{\bar{t}}=1} y_{\bar{t}} \mathscr{A}_{\bar{t}}) \right) \tag{8}$$

That is, the availability of the VDC $\mathscr{A}_{vdc}$ is at least $\mathscr{A}^{lb}_{vdc}$.

3.2 Availability-Aware Embedding

In the following we formulate the availability-aware embedding problem. We start by describing the embedding constraints then provide the optimization objective function.

- **Embedding constraints:**

When embedding the VDC, there are many constraints that should be satisfied. For instance, in order to ensure that the capacities of physical resources are not violated, the following constraints must be satisfied:

$$\sum_{n \in N'} x_{n\bar{n}} c^r_n \leq c^r_{\bar{n}} \quad \forall \bar{n} \in \bar{N}, r \in R \tag{9}$$

$$\sum_{l \in L'} f_{l\bar{l}} b_l \leq b_{\bar{l}} \quad \forall \bar{l} \in \bar{L} \tag{10}$$

We also require the link embedding to satisfy the flow constraint between every source and destination node pairs in each VDC topology, namely:

$$-\sum_{\bar{l} \in \bar{L}} \bar{v}_{\bar{n}\bar{l}} f_{l\bar{l}} + \sum_{\bar{l} \in \bar{L}} \bar{u}_{\bar{n}\bar{l}} f_{l\bar{l}} = \sum_{n \in N} x_{n\bar{n}} u_{nl} - \sum_{n \in N} x_{n\bar{n}} v_{nl} \quad \forall l \in L', \bar{n} \in \bar{N} \tag{11}$$

Here $\sum_{n \in N} x_{n\bar{n}} u_{nl}$ is equal to 1 if n is the source of the link l of VDC and n is embedded in the physical node $\bar{n}$. Equation (11) essentially states that the total outgoing flows of a physical node $\bar{n}$ for a virtual link l is equal to the total incoming flows unless $\bar{n}$ hosts either a source or a destination virtual node.

Furthermore, we need to consider the node placement constraints. For instance, VMs should only be placed in physical servers whereas virtual switches may be placed either in switches (e.g., flowvisor instance [18]) or servers (e.g., open vSwitch instance [19]). Hence, we define $\tilde{x}_{n\bar{n}}$ to indicate whether physical node $\bar{n}$ is able to host virtual node n of the VDC. Thus, if a virtual node n is a virtual machine (not a switch), we have $\tilde{x}_{n\bar{n}} = 0 \, \forall \bar{n} \in \bar{S}$ and $\tilde{x}_{n\bar{n}} = 1 \, \forall \bar{n} \in \bar{M}$. Hence, the the placement constraint can be written as:

$$x_{n\bar{n}} \leq \tilde{x}_{n\bar{n}} \quad \forall n \in N', \bar{n} \in \bar{N} \tag{12}$$

Additionally, we need to ensure that the minimum number of provisioned virtual nodes is at least the number of nodes required by the SP. Hence, we have:

$$\sum_{n \in N'} \sum_{\bar{n} \in \bar{N}} x_{n\bar{n}} \geq |N| \tag{13}$$

Furthermore, to ensure that the virtual switch is mapped, the following equation must hold:

$$\sum_{\bar{n} \in \bar{N}} x_{0\bar{n}} = 1 \tag{14}$$

Furhermore, $y_{\bar{n}}$ must be equal to 1 if the physical node $\bar{n}$ is used to host any virtual node or used as an intermediate node to embed a virtual link. This implies the following constraints must hold:

$$y_{\bar{n}} \geq x_{n\bar{n}} \quad \forall n \in N', \bar{n} \in \bar{N} \tag{15}$$

$$y_{\bar{n}} \geq w_{n\bar{n}} \quad \forall n \in N', \bar{n} \in \bar{S} \tag{16}$$

$$y_{\bar{n}} \geq f_{l\bar{l}} \bar{u}_{\bar{n}\bar{l}} \quad \forall \bar{n} \in \bar{N}, l \in L', \bar{l} \in \bar{L} \tag{17}$$

$$y_{\bar{n}} \geq f_{l\bar{l}} \bar{v}_{\bar{n}\bar{l}} \quad \forall \bar{n} \in \bar{N}, l \in L' \tag{18}$$

We have also to ensure that the VDC availability $\mathscr{A}^{lb}_{vdc}$ is higher than the required availability. That is:

$$\mathscr{A}^{lb}_{vdc} \geq \mathscr{A} \tag{19}$$

- **Objective function:**

The goal of the embedding is to minimize the number of the physical nodes used for embedding the VDC as well as the amount of consumed bandwidth while maintaining the constraints of Eqs. (9)–(19). Hence our objective function can be written as follows:

$$\min\left(\alpha\sum_{\bar{n}\in\bar{N}} y_{\bar{n}}p_{\bar{n}}+\beta\sum_{\bar{l}\in\bar{L}}\sum_{l\in L'} f_{\bar{l}l}b_l+\gamma\sum_{\bar{n}\in\bar{N}}\sum_{n\in N'}\left(x_{n\bar{n}}\sum_{r\in R} w^r c_n^r\right)\right) \tag{20}$$

where $p_{\bar{n}}$ is the energy cost defined as:

$$p_{\bar{n}} = \begin{cases} 0 & \text{if the machine } \bar{n} \text{ is already active} \\ t_{c\bar{n}} & \text{if the machine is off} \end{cases} \tag{21}$$

$t_{c\bar{n}}$ is cost of turning on the machine $\bar{n}$ and w^r is the weight factor for resource type $r \in R$, which depends on the scarcity of the resource. The weight factor α, β and γ are used to strike the balance between energy cost, communication cost, and computation cost. This optimization problem is $\mathcal{NP}$-hard as it generalizes the multi-dimensional bin packing problem [4]. Therefore, we provide a heuristic to solve the problem in the subsequent section.

4. Heuristic

This section describes a heuristic for solving the availability-aware VDC embedding problem that ensures that each embedded VDC satisfies its requirements in terms of availability and resources. The goal is to minimize the number of active machines and the consumed bandwidth in the data center network with the goal of increasing CP's income.

Our algorithm is described in Algorithm 1. The VDC embedding is carried out in two phases: (1) VM mapping, (2) virtual switch and link mapping. All physical servers are sorted based on their status (active or inactive) and availability. Since our aim is to reduce the number of active servers, the algorithm tries first to embed VMs in the active servers. When a VDC is received, the VMs are sorted in descending order according to size of their requested resources. The size of VM n is captured by $size_n$ defined as:

$$size_n = \sum_{r\in R} w^r c_n^r \quad \forall n \in N \tag{22}$$

where w^r is the weight factor for resource type $r \in R$, which depends on the scarcity of the resource. The intuition is that it is usally harder to map large VMs. The algorithm parses the sorted list of servers and finds the one that can satisfy the resource requirements of the VM n. This aims also at embedding VMs in servers with the highest availability in order to avoid the need for backups. After embedding all VMs $n \in N$, we start mapping the virtual switch and the links. If $\mathscr{A}_{vdc} < \mathscr{A}$, we provision B backups where B is the minimum number of VMs that are embedded in a single physical server. That is:

$$B = \min\left(\sum_{n\in N} x_{n\bar{n}}\right) \quad \forall \bar{n} \in \bar{N} \tag{23}$$

The idea is to provision enough backups to take over the failure of the physical machines hosting the lowest number of VMs. After embedding B backup VMs, we check again $\mathscr{A}_{vdc}$. If it is still lower than the requested availability, another VM backup is provisioned in a new physical node that is not hosting any of the previously embedded VMs. This process is repeated until the required availability is reached. In this case, the VDC is accepted. In order to avoid overly backup provisioning, the CP can set a maximum number of

Algorithm 1 Availability-aware VDC embedding

```
1: VM Mapping Phase:
2: Sort servers M̄ by status (active or not) and availability
3: Sort N by their size_n^i defined in Eq. (22)
4: for each virtual machine n ∈ N do
5:     if EmbedNode(n, 1) = −1 then
6:         Reject request
7:     end if
8: end for
9: B ← 0
10: while A_vdc^lb < A and B ≤ B_max do
11:     if B = 0 then
12:         B = (min_{n̄∈N̄} Σ_{n∈N} x_{nn̄})
13:         EmbedNode(n_max, B) {n_max: the node with the largest
            size.}
14:     else
15:         B ← B + 1
16:         EmbedNode(n_max, 1)
17:     end if
18: end while
19: Switch and Link Mapping Phase:
20: C_c,min ← ∞ {C_c,min is the minimum communication cost}
21: for each n̄ ∈ N̄ do
22:     cost(n̄) ← 0
23:     Compute total communication cost C_c
24:     if C_c < C_c,min then
25:         C_c,min ← C_c ; p ← n̄ {p: candidate physical node for
            hosting the switch}
26:     end if
27: end for
28: if B ≤ B_max then
29:     Accept the VDC request
30: else
31:     Reject the VDC request
32: end if
```

Algorithm 2 EmbedNode(n,b)

```
1: Input : virtual node n, number of replicas b
2: Output: Physical node n̄ or -1
3: Sort servers M̄ by status (active or not) and availability
4: for i ← 1, b do
5:     Find n̄ ∈ M̄ able to host n
6:     if Not found then
7:         Return -1
8:     end if
9:     Embed n in n̄
10: end for
11: Return n̄
```

backups $B_{\max}$. If the number of backups exceeds $B_{\max}$, the request is rejected.

Once the VM mapping is done, the virtual switch and link mapping are carried out jointly. To embed a virtual link, we consider the shortest path between the physical node hosting the switch and the one hosting a VM. We define the virtual link communication cost as the total number of hops in the corresponding physical path multiplied by the link bandwidth (i.e., $hop\_count \times bandwidth$). The total VDC communication cost (C_c) is then defined as the sum of communication costs of all virtual links. In order to find the optimal embedding for the virtual switch and the links, the algorithm computes the VDC communication cost for all possible placements of the virtual switch. The final embedding is the one that minimizes this cost.

In our heuristic, we first try to embed VMs into physical servers that are already active. This leads to less energy consumption, and hence to reduced costs. Furthermore, we try to embed VMs of the same VDC as close as possible to each other. This reduces communication costs between VMs and the consumed bandwidth in the network. Finally, the algorithm adds backup incrementally until the required availability is met to avoid the over-estimation of backup requirement. Therefore, our heuristic takes into consideration the goals stated in the objective function (Eq. (20)).

5. Experiments

In this section, we evaluate the effectiveness of our availability-aware VDC embedding algorithm (Hi-VI) through simulations. To this end, we simulate a physical data center of 120 physical machines organized into four racks. The data center network consists of 4 top-of-rack switches, 4 aggregation switches, and 4 core switches connected according to the VL2 topology. We assume each physical machine contains 4 CPU cores, 8 GB of memory, 1 TB hard disk space, and 1 Gbps NIC cards. In order to consider the heterogeneity of the data center equipments, availabilities of servers and switches are selected randomly from $\{0.99, 0.999, 0.9999, 0.99999\}$. VDC requests arrive following a Poisson distribution with an average rate of 0.02 requests per second during day time and 0.01 requests per second during night time. This reflects demand fluctuations in data centers. We assume all VDCs have a star topology consisting of a single virtual switch connected to multiple VMs. The number of VMs per VDC is taken randomly between 1–20. The size of each VM in terms of CPU, memory and disk is chosen randomly between $1-4$ cores, $1-2$ GB of RAM and $1-10$ GB of disk space, respectively. The capacity of virtual links are also generated randomly between 1–100 Mbps. Furthermore, the required availability for each VDC is generated randomly from $\{0.99, 0.999, 0.9999\}$ chosen purposely to be higher than the availability guaranteed by Google Apps SLA [20]. The lifetimes of VDCs are exponentially distributed with an average of 3 hours. Finally, if a VDC is not accepted immediately because it is not possible to meet the requirements in terms of availability or resources, it is kept in a waiting queue for a maximum of one hour after which it is automatically withdrawn.

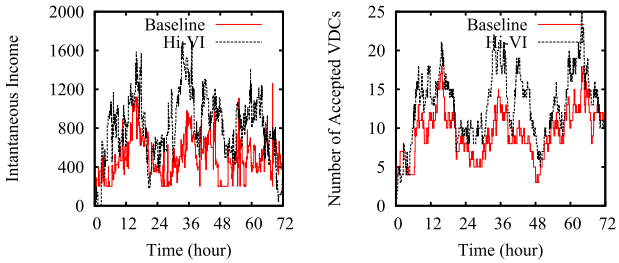
Since, previous proposals in the literature ignore equipment heterogeneity in production data centers, it is not possible to directly compare them to Hi-VI. Therefore, we developed a baseline resource allocation algorithm that combines [11] and [16]. Specifically, the baseline operates in two steps: similar to [11], it starts by spreading VMs across active physical nodes in order to maximize the availability. Then, the algorithm provisions a backup VM in a randomly selected physical node and evaluates the VDC availability. This backup provisioning process is repeated until the required availability is satisfied. It is worth noting that, similar to [16], the baseline is oblivious to the existent heterogeneity in terms of failure rates and availability of the underlying physical components (since the placement of backups does not consider the availability of physical nodes).

We first evaluate the instantaneous income of Hi-VI and the number of accepted VDC requests compared to the baseline algorithm. The instantaneous income is provided by the following equation[†]:

$$\mathcal{R}_{inst} = \sum_{v \in V} \left(\mu^b \sum_{l \in L} b_l^v + \sum_{n \in N} \sum_{r \in R} \mu^r c_n^{vr} \right) - \mu^e \sum_{\bar{n} \in \bar{N}} y_{\bar{n}} E_{\bar{n}} \quad (24)$$

where μ^b and μ^r are the unit selling prices for bandwidth and resource type r for a single timeslot, respectively. V is the set of embedded VDCs at the current timeslot and the superscript v refers to VDC number v. The energy cost paid by the CP and the energy consumed by machine $\bar{n}$ during a timeslot are denoted by μ^e and $E_{\bar{n}}$, respectively.

Figure 2(a) shows that Hi-VI leads to much higher instantaneous income than the baseline. Figure 2(b) confirms that our algorithm accepts more VDC requests than the baseline. One reason for this higher income is the higher acceptance of VDC requests. Another reason is that the number of used physical machines is higher with the baseline algorithm than with Hi-VI (Fig. 3). This is because the baseline algorithm spreads the VMs across the physical machines, and hence turns on more servers. Thus, it leads to a higher energy costs than Hi-VI. The utilization of the dif-



(a) Instantaneous Income (b) Number of Accepted VDC

Fig. 2 Income and number of embedded VDCs.

[†]In order to compute the instantaneous income, resource demands (in terms of CPU, memory and bandwidth) are normalized between 0 and 1.

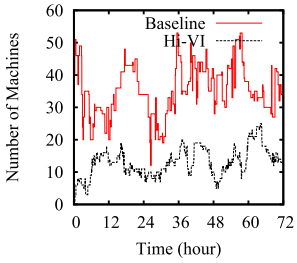
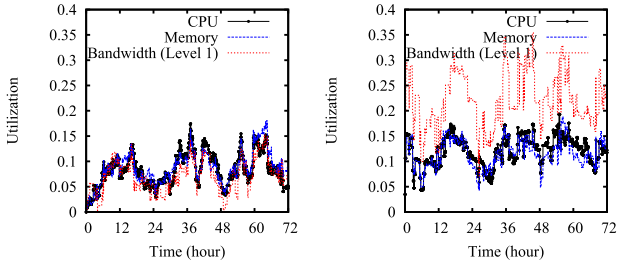
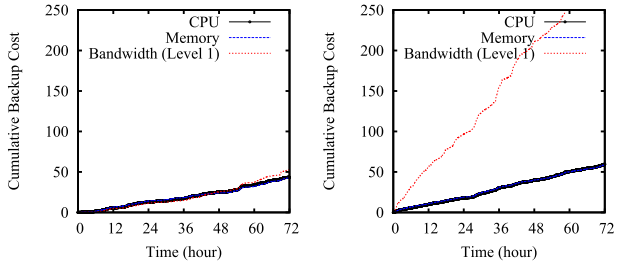


Fig. 3 Number of active physical machines.



(a) Hi-VI (b) Baseline

Fig. 4 Utilization of CPU, memory and bandwidth.



(a) Hi-VI (b) Baseline

Fig. 5 Backup cost.

ferent resources (CPU, memory, and bandwidth) for Hi-VI and the baseline is depicted in Fig. 4. We can notice that the utilization of CPU and memory are comparable for both algorithms. However, the baseline has accepted much less VDCs, which means that the baseline has allocated a lot of resources for provisioning the backups. We also notice a significant difference in the bandwidth utilization. The baseline requires more bandwidth than Hi-VI, although it accepts less VDC requests. The baseline spreads the VMs across the physical servers and thus uses more bandwidth to embed the virtual links. Finally, Fig. 5 shows the cumulative backup costs of CPU, memory and bandwidth for both algorithms. The backup cost of a VDC is computed as the amount of resources used by the provisioned backup multiplied by its lifetime. We can see that the baseline has allocated much more backup resources than Hi-VI in terms of cpu, memory and bandwidth. These results clearly show that Hi-VI outperforms the baseline in terms of income, accepted VDC requests and significantly reduces backup costs.

6. Conclusions

As resource availability is a prime concern for cloud users, providers are prompted to roll out computing and networking resources with more stringent availability guarantees. Despite recent research on the problem, none has considered the heterogeneity of the production data center components in terms of failure rates and availability to estimate the required amount of backup resources reserved to ensure the targeted availability.

In this paper we proposed Hi-VI, a VDC management framework that takes into account the heterogeneity of cloud data center equipments to dynamically provision backup resources in order to ensure required VDC availability. Through simulations, we demonstrated that Hi-VI is able to satisfy VDC's availability and resource requirements while minimizing operational costs (notably energy costs). Compared to heterogeneity-oblivious solutions, Hi-VI increases by up to 20% the cloud provider net income while minimizing by up to 40% the operational costs.

Acknowledgment

This work was supported by the Natural Science and Engineering Council of Canada (NSERC) under the Smart Applications on Virtual Infrastructure (SAVI) Research Network.

References

[1] Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/ec2/
[2] Google Compute Engine. https://cloud.google.com/
[3] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards predictable datacenter networks," Proc. ACM SIGCOMM, Aug. 2011.
[4] M.F. Zhani, Q. Zhang, G. Simon, and R. Boutaba, "VDC Planner: Dynamic migration-aware virtual data center embedding for clouds," IM, 2013.
[5] M. Rabbani, R. Esteves, M. Podlesny, G. Simon, L.Z. Granville, and R. Boutaba, "On tackling virtual data center embedding problem," IM, 2013.
[6] C. Guo, G. Lu, H.J. Wang, S. Yang, C. Kong, and P. Sun, "Secondnet: A data center network virtualization architecture with bandwidth guarantees," ACM CoNEXT, 2010.
[7] Downtime Outages and failures, understanding their true costs. http://www.evolven.com/blog/downtime-outages-and-failures-understanding-their-true-costs.html
[8] X. Wu, D. Turner, C.C. Chen, D.A. Maltz, X. Yang, L. Yuan, and M. Zhang, "Netpilot: Automating datacenter network failure mitigation," SIGCOMM Comput. Commun. Rev., vol.42, no.4, Aug. 2012.
[9] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: Measurement, analysis, and implications," Proc. ACM SIGCOMM, 2011.
[10] J. Xu, J. Tang, K. Kwiat, W. Zhang, and G. Xue, "Survivable virtual infrastructure mapping in virtualized data centers," IEEE International Conference on Cloud Computing (CLOUD), 2012.
[11] P. Bodík, I. Menache, M. Chowdhury, P. Mani, D.A. Maltz, and I. Stoica, "Surviving failures in bandwidth-constrained datacenters," Proc. ACM SIGCOMM, 2012.

[12] W.L. Yeow, C. Westphal, and U.C. Kozat, "Designing and embedding reliable virtual infrastructures," Tech. Rep., March 2010.
[13] A. Greenberg, J.R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D.A. Maltz, P. Patel, and S. Sengupta, "VL2: A scalable and flexible data center network," Proc. ACM SIGCOMM, 2009.
[14] K.V. Vishwanath and N. Nagappan, "Characterizing cloud computing hardware reliability," Proc. ACM Symposium On Cloud Computing, SOCC, 2010.
[15] H. Yu, V. Anand, C. Qiao, and G. Sun, "Cost efficient design of survivable virtual infrastructure to recover from facility node failures," IEEE ICC, pp.1–6, 2011.
[16] W.L. Yeow, C. Westphal, and U.C. Kozat, "Designing and embedding reliable virtual infrastructures," SIGCOMM Comput. Commun. Rev., vol.41, no.2, April 2011.
[17] M. Rahman and R. Boutaba, "SVNE: Survivable virtual network embedding algorithms for network virtualization," pp.1–14, 2013.
[18] R. Sherwood, G. Gibb, K.K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "Can the production network be the testbed?," Proc. USENIX Conference on Operating Systems Design and Implementation, OSDI, 2010.
[19] Open vSwitch. http://openvswitch.org/
[20] Google Apps Service Level Agreement. http://www.google.com/apps/intl/en/terms/sla.html

Md Golam Rabbani is currently pursuing his M.Math. degree in Computer Science at University of Waterloo, under the supervision of Prof. Raouf Boutaba. He received the Master of Information Technology degree from Monash University, Australia, in 2011, and the B.Sc. degree in Computer Science and Engineering from Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2007. He worked as a system engineer in a telecommunication company from 2007 to 2009. His research interests include data center, cloud computing, future Internet architecture, and wireless communication.

Mohamed Faten Zhani is currently a postdoctoral research fellow at the University of Waterloo, Canada (advisor: Raouf Boutaba). He received his Ph.D. in Computer science from the University of Quebec in Montreal, Canada in 2011. His research interests include virtualization, resource management in cloud computing environment and network performance evaluation. Web page: https://cs.uwaterloo.ca/~mfzhani/

Raouf Boutaba received the M.Sc. and Ph.D. degrees in computer science from the University Pierre & Marie Curie, Paris, in 1990 and 1994, respectively. He is currently a professor of computer science at the University of Waterloo and a distinguished visiting professor at the division of IT convergence engineering at POSTECH. His research interests include network, resource and service management in wired and wireless networks. He is the founding editor in chief of the IEEE Transactions on Network and Service Management (2007–2010) and on the editorial boards of other journals. He has received several best paper awards and other recognitions such as the Premiers Research Excellence Award, the IEEE Hal Sobol Award in 2007, the Fred W. Ellersick Prize in 2008, and the Joe LociCero and the Dan Stokesbury awards in 2009. He is a fellow of the IEEE.