future internet

MDPI

Article

v-Mapper: An Application-Aware Resource Consolidation Scheme for Cloud Data Centers †

Aaqif Afzaal Abbasi [1,*] and Hai Jin [2,*]

[1] School of Computer Science, Wuhan University, Wuhan 430072, China
[2] Services Computing Technology & System Laboratory, Cluster & Grid Computing Laboratory, Big Data Technology & System Laboratory, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China
* Correspondence: aaqif@whu.edu.cn (A.A.A.); hjin@hust.edu.cn (H.J.); Tel.: +86-27-8754-3529 (ext. 8033) (A.A.A.); Fax: +86-27-8755-7354 (A.A.A.)
† This paper is an extended version of paper published in Proceedings of the 9th Asia-Pacific Services Computing Conference (APSCC'15), Bangkok, Thailand, 7–9 December 2015.

Received: 21 July 2018; Accepted: 12 September 2018; Published: 15 September 2018

Abstract: Cloud computing systems are popular in computing industry for their ease of use and wide range of applications. These systems offer services that can be used over the Internet. Due to their wide popularity and usage, cloud computing systems and their services often face issues resource management related challenges. In this paper, we present v-Mapper, a resource consolidation scheme which implements network resource management concepts through software-defined networking (SDN) control features. The paper makes three major contributions: (1) We propose a virtual machine (VM) placement scheme that can effectively mitigate the VM placement issues for data-intensive applications; (2) We propose a validation scheme that will ensure that a cloud service is entertained only if there are sufficient resources available for its execution and (3) We present a scheduling policy that aims to eliminate network load constraints. We tested our scheme with other techniques in terms of average task processing time, service delay and bandwidth usage. Our results demonstrate that v-Mapper outperforms other techniques and delivers significant improvement in system's performance.

Keywords: network; systems; cloud computing; data center; performance; software-defined; virtual machine; scheduling; admission control; application-aware

1. Introduction

Cloud computing is taking the computing world by storm. It has the potential to transform a large part of the IT industry, making software even more attractive as a service and shaping the way IT hardware is designed and purchased [1,2]. One of the big promises of cloud computing is its ability to provide high quality services at a low cost, which is a result of sharing resources among many users. European Commission outlined actions to deliver a net gain of millions of new jobs over next years. Despite the barriers to wider adoption, cloud technology is a crucial driver for growth in European markets [2]. However, the actual ability to provide these services at a low cost critically depends on the efficiency of the resource use in the cloud. Despite attaining maturity in many major areas of service provisioning, cloud computing is still developing.

In general, there are two serious issues in deploying and provisioning resources in cloud environments, namely virtual machine (VM) placement and lack of precise task scheduling schemes. These schemes also influence resource provisioning strategies for multimedia-based dynamic cloud services [3,4]. Refined resource allocation is usually implemented by deploying VM instances and

customizing their resources on demand, but this impacts the performance of VMs in completing customers' workloads. Refined resource allocation involves the management of multiple types of resources, such as the CPU, memory and I/O devices. VM placement and scheduling are considered to be critical components of cloud computing because they directly affect a provider's performance. Network operators are responsible for configuring their network resources to enforce various high-level policies and to enable them to respond to a wide range of network events. These policies also focus on preserving the costs of data centre management [5]. Therefore, whenever the demand for network resources increases, network operators have to deal with resource allocation and its related challenges [5,6].

Figure 1 illustrates some of the many cloud services and applications provided by data centers. The question of how to design an appropriate resource management scheme that can satisfy both providers and customers is becoming a major issue in modern cloud environments. Monitoring and management services in cloud systems are a fundamental concern. The primary reason is the continuously changing availability of system resources and needs [1,2]. It is, therefore, important to address resource provisioning concerns for relevant cloud services and technologies in an organized manner. One potential solution is that cloud vendors could offer specialized hardware and software techniques in order to deliver higher reliability, albeit this would presumably come at a higher price. This reliability could then be sold to users as part of a service level agreement. The high-availability computing community has long followed the mantra "no single point of failure", yet in terms of reliability, the management of a cloud computing service by a single company is in fact a potential single point of failure and thus a reliability concern [1,2]. Even if a company has multiple data centers in different geographic regions using different network providers, it may have common software infrastructure and accounting systems that are at risk, or the company could even go out of business with the network going down.

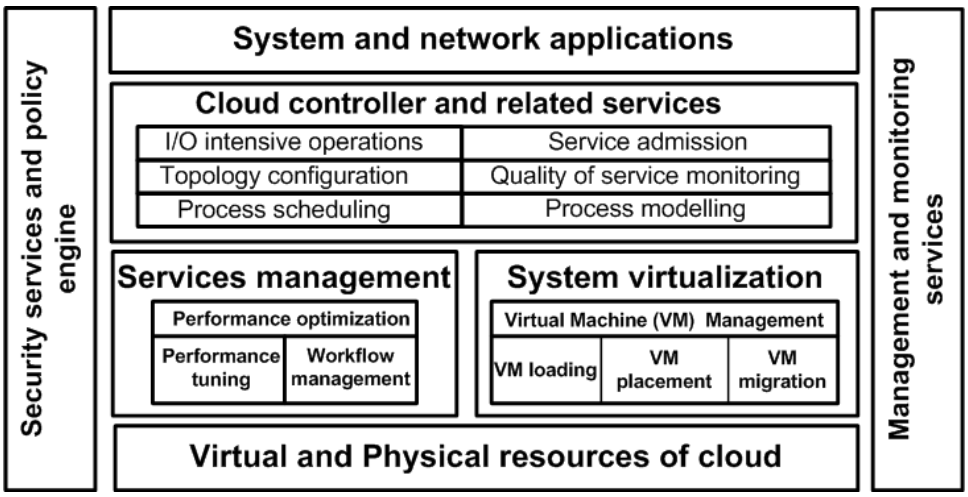

Figure 1. An overview of cloud services and applications.

Today's networks have limited capacity to answer modern network reliability issues. Network operators are trying to implement network policies using command line interface (CLI) environment [7,8]. Unfortunately, mainstream operating systems are unaware of synchronization operations within multithreaded applications. This could significantly aggravate both the system throughput and fairness. In addition, network states change continually, and at present, operators must manually adjust their network configuration in response to the changing network conditions [9,10].

To overcome cumbersome resource allocation challenges, the present paper proposes, presents and evaluates a novel paradigm in the field of cloud resource management named v-Mapper. v-Mapper employs applied graph theory and Fuzzy Analytical Hierarchy Process (Fuzzy-AHP) techniques to optimize network performance behavior. This helps in providing a clean representation of VMs on a

cloud mesh. It enables system administrators to fully realize the potential of application-awareness concepts by putting them under automated control and help service providers to deliver best service packages to small and medium enterprises (SMEs) [11].

The specific aim of this paper is to establish the necessary baseline for a tool-supported decision support method aimed at facilitating the selection of cloud services in a multi-cloud environment. Herein, we propose v-Mapper, a new scheme that implements application-aware features to oversee VM placement, services admission and task scheduling functions in a cloud environment.

The rest of this paper is organized as follows. Section 2 discusses the related work in conventional VM management approaches. Section 3 presents background information on application-awareness and related concepts in existing cloud data centers. In Section 4, we elaborate v-Mapper's functions and features in detail. Section 5 provides a comprehensive discussion on the test bed and simulation settings. In Section 6, we present the performance evaluation results and carefully evaluate our solution. Section 7 concludes this paper and discusses the future work.

2. Related Work

In the cloud environment, resources must be shared in such a way that a user's applications do not affect other users' applications. Therefore, the resources of all the users have to be partitioned so that they are private and secure. Our research is related to cloud resource management. In the following section, we highlight some important work performed in cloud resource administration functions.

2.1. VM Management in Cloud Datacenters

Applications in cloud infrastructures acquire virtual machines (VMs) from providers when necessary. The current interface for acquiring VMs from most providers, however, can be too limiting for tenants. Data-intensive applications [11–13] need to communicate with datacenters frequently, and therefore, the traffic loads among these VMs are particularly heavy. This can also increase system overhead and potentially degrade the network performance. Energy efficient placements seek to consolidate VMs for resource use, which can also greatly impact network performance. This can lead to situations in which VM pairs are placed on host machines whose traffic volumes are already large.

VMs typically tend to follow patterns in their resource demands. A study [14] of a large number of CPU traces from production servers showed that most of the demand traces had a high correlation with a periodic behavior. Statistical multiplexing can also be used to exploit possible correlations among VMs. A continuous monitoring approach to study VM resource usage patterns on all running VMs has been presented in [15,16].

In recent years, VM placement concepts have been exploited to address important data center infrastructure related challenges. However, supporting tenant-controlled VM placement makes it difficult, if not impossible, to perform simple resource management tasks. We believe that if application-awareness concepts were to be employed, it would help provide a simple and clean interface for datacenter administrators.

2.2. SDN Role in Datacenter Optimization

Datacenters not only provide a flexible and reliable storage space but also support the underlying virtualization infrastructure. SDN solutions focus on the need for the use of programmability in the data center network by encouraging the implementation of comprehensive management capabilities that can conclusively demonstrate operation compliance. This helps networks to reap several benefits, such as cost reduction and ease of management [17]. SDNs are reshaping the future of networks by changing the behavior of conventional network operations. An SDN's management plane can be seen as a management-cum-control platform that accommodates traditional network management services and protocols. The Open Network Foundation (ONF) workgroup on Configuration and Management encourage network system developers to propose network solutions such as Procera [7], Frentic [18] and Meridian [19] which encourage the usage of SDN-based solutions.

3. Application-Awareness Concepts in Cloud Datacenters

Conventional networking-based cloud resource administration techniques fail to address the dynamicity in the networking arena, especially ones that require a lot of human intervention to provide the network resources in an on-demand manner. Software-defined networking is a key factor driving the software-defined cloud (SDC) research community. Figure 2 presents a typical software-defined cloud environment. To effectively manage the middlebox processing capabilities in SDCs, the concept of outsourcing enterprise middlebox processing to the cloud was proposed in [20] with a system the authors called APLOMB (Appliance for Outsourcing Middleboxes). APLOMB delegates middlebox tasks to the cloud to ensure the ease of management and capital expenditure (CAPEX) functions. In multi-tenant datacenters, tenants need to perform migration of unmodified workloads from enterprise networks to the service provider networks. This is a very challenging task and was addressed by Network Virtualization Platform (NVP) technology in [21]. The proposed solution uses SDN to decouple network management from the physical infrastructure in two steps; first, by decoupling the tenant control planes from the physical infrastructure, and then by decoupling the modified workload of NVP from the physical infrastructure.

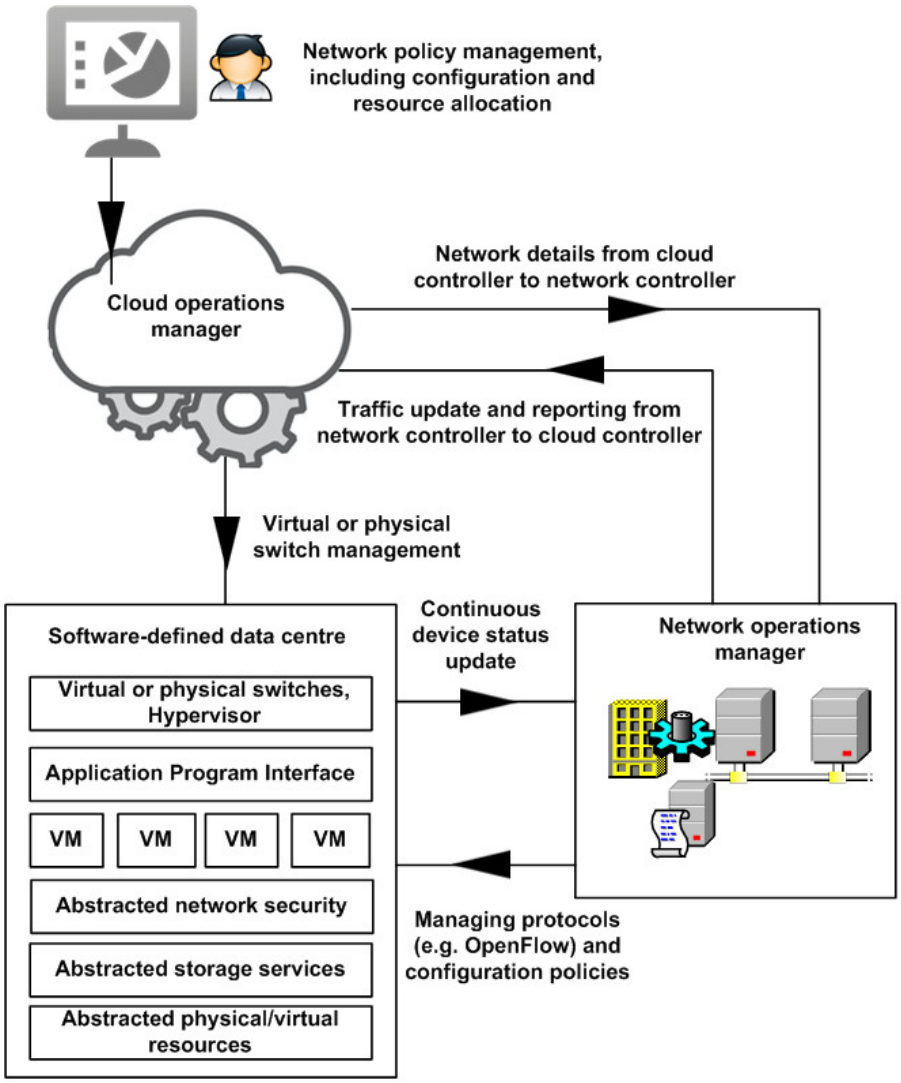

Figure 2. A view of the services and functions in a software-defined cloud data center.

There is a dire need to support the easy development of software-defined cloud environments. In this regard, a recent data centre solution, Topology and Orchestration Specification for Cloud Applications (TOSCA) [22] has been presented. TOSCA also provides a flexible combination of declarative and imperative models of workloads that can be seamlessly integrated with automation frameworks for task-level automation and optimized resource orchestration. TOSCA was exclusively designed to overcome the interoperability challenges across the cloud and virtualization infrastructure and to simplify the management and deployment of applications in cloud datacenters. To effectively monitor service-centric cloud environments, a unified solution was presented in [23], which can rapidly migrate and optimize the use of cloud resources in ways that are consistent with the service profiles and policies. This solution also address interoperability concerns for various cloud service platforms. The reliable implementation of application-aware concepts through SDN implementation is now an essential requirement for leading data center solutions. We believe that a combination of open standards technology (SDN technology) with a best-of-breed hardware platform (SDN-enabled hardware peripherals) can deliver flexible, scalable and high performance networking solutions for the future networking industry.

4. Data Center Design and the v-Mapper System Model

Below we describe the v-Mapper model and its features in detail. A list of symbol notations is provided in Table 1.

Table 1. Notations list.

Notation	Description
SN	Storage Node
DP	Data Packets
U	Evaluation index
V	Set of comments
R	Evaluation matrix
FW	Factor Weight
P	Probability
SR	Shared Resources (Available)
SFL	Shared Fair Load
Q_{req}	Scheduling Request Query
R_t	System Threshold Capacity
B_r	Buffered requests
P_{qi}	Priority value
B	Adjacency matrix between storage and cloud node matrices as i and j
N	Number of aggregate compute nodes
S_R^{SFL}	Shared fair load SFL of the shared resource SR

4.1. VM Placement and the Data Center Model

Modern data center architecture designs rely on the path multiplicity to achieve horizontal scaling of hosts [23]. Therefore, data center topologies are very different from conventional enterprise network topologies. Data center topologies employ various forms of multi-rooted spanning trees to address congestion-related constraints. For the development of our model, we started by modeling a data center network and its application jobs as a static problem. We then formulated a joint VM placement and scheduling strategy that aimed to minimize these constraints.

In our model, we consider a cloud environment consisting of two storage nodes SN1 and SN2 and three compute nodes CN1, CN2 and CN3. We also use three data packets DP1, DP2 and DP3. The modeling scheme presented is similar to that presented in [24], except for the fact that we consider a graph theoretic representation for our cloud architecture. The presented cloud scenario is mapped in Figure 3.

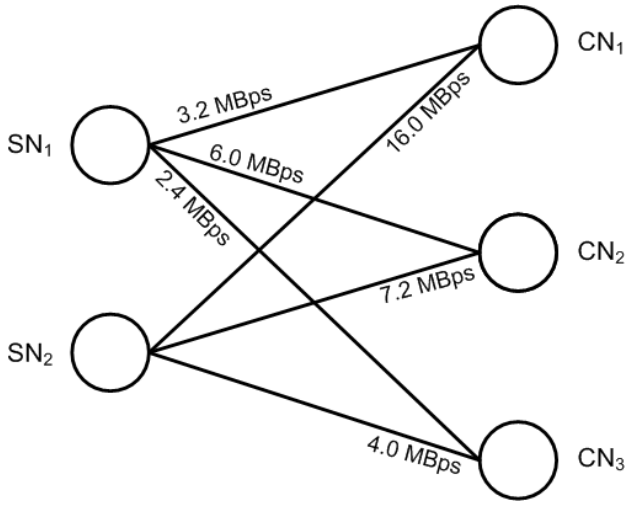

Figure 3. CN-SN relationship as a bipartite.

We then interpret the nodes and edges relationship (in Figure 3) by developing an adjacency matrix:

$$B = \left[b_{ij}\right]_{n\times m} = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1m} \\ b_{21} & b_{22} & \cdots & b_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nm} \end{bmatrix} \tag{1}$$

By populating data in the adjacency matrix (the same way as mentioned in [24]), we obtain the following matrix:

$$B_{3,2}\begin{bmatrix} 3.2 & 16.0 \\ 6.0 & 7.2 \\ 2.4 & 4.0 \end{bmatrix} \text{MBps} \tag{2}$$

By loading the data packets (DPs) on the storage nodes (SNs), the matrix can be represented as:

$$D_{2,2} = [d_{ki}]_{2\times 2} = \begin{bmatrix} DP_1 & DP_3 \\ DP_2 & 0 \end{bmatrix} = \begin{bmatrix} 200 & 500 \\ 100 & 0 \end{bmatrix} \text{MB} \tag{3}$$

The response time R of an individual node can be calculated as:

$$\left[\tau_{ij}\right]_{n\times m} = \left[\frac{1}{b_{ij}}\right]_{n\times m} \text{ where } 1 \leq i \leq n \text{ and } 1 \leq j \leq m \tag{4}$$

Therefore, the total response time for a CN can be calculated by using the following expression:

$$t_{CN,\,i} = \sum_{j=1}^{n} t_{R,\,ji} \tag{5}$$

We briefly present our VM placement scheme in Algorithm 1.

The presented VM placement algorithm is different from its classic predecessors in a way that it employs Fuzzy AHP on a graph theoretic model. This combinatorial approach makes it lightweight and efficient for simple VM placement solutions with limited number of cloud and storage nodes. However, this approach may not be suitable for larger networks because of complexities in resource allocation and time variation of applications. To manage the offline VM allocation, a requests buffer

has been proposed which can queue the waiting buffered requests and provide resources to VMs on demand.

Algorithm 1. VM placement

Input: Load DP on SN
Output: VM placement decision
Let CN denote Compute Node of a cloud network
Let $t_{CN,i}$ denote service response time of a CNi
Let least denote least response time of all CN
Let j denote the CN with least response time
Calculate t_R for each CN as
$t_{CN,i} = \sum_{j=1}^{n} t_{R,ji}$
$i \leftarrow 0$
$j \leftarrow 0$
least $\leftarrow t_{CN,0}$
while I < n do
  if least > $t_{CN,i}$ then
    least $\leftarrow t_{CN,i}$
    $j \leftarrow i$
  end if
  $i \leftarrow i + 1$
end while
Place VM on CN j
Exit

4.2. VM Workload Admission

VM resource provisioning plays a key role in ensuring that cloud providers adequately accomplish their obligations to customers, while maximizing the use of the underlying infrastructure. A capable workload admission scheme requires dynamically allocating each service request with the minimal resources that are needed for acceptable fulfillment of that service requirement, leaving the surplus resources free to deploy more VMs. The decision-making process for workload admission in v-Mapper takes place through a fuzzy comprehensive evaluation [25]. The process involves several steps, as described below.

(1) First, we determine the factor and sub-factor weights for cloud services through an evaluation index (U). The factor and sub-factor weights are assigned through priority criteria of AHP. A simple case example is presented in [1] and can be used to measure functions influencing a cloud service's resource management concerns.
(2) "Not all clouds are created equal"; therefore, we create a set of comments (V) to describe the evaluation of cloud services by using phrases such as "Acceptable", "Constrained", etc. These are determined based on Saaty's 1 to 9 scale [26] to describe the preferences between alternatives as being either equally, moderately, strongly, very strongly or extremely preferred.
(3) To provide a comprehensive assessment methodology, we create an evaluation matrix (R) from U to V, where each factor u_i ($i \leq n$) can be written as a fuzzy vector $R_i \in \mu(V)$. Mathematically, this fuzzy relationship can be expressed as:

$$R = \left(r_{ij}\right)_{nxm} = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1m} \\ r_{21} & r_{22} & \cdots & r_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n1} & r_{n2} & \cdots & r_{nm} \end{pmatrix} \quad (6)$$

The evaluated result of Equation (6) should match the normalized conditions, because the sum of the weight of the vector is 1 (i.e., for i, $r_{i1} + r_{i2} + r_{i3} + \ldots \ldots .. + r_{im} = 1$).

(4) A factor assigned to a number in a computation system reflects its importance. The greater the weight of a factor is, the more importance it has in the system. We, therefore, determine the factor weight (FW) of each factor in the evaluation index (U) system.

(5) We obtain the evaluation result (E) through the product of the factor weight (FW) and the evaluation matrix (R). This can be denoted as E = FW(R) = (E1, E2, E3, ... , Em). Finally, the evaluated weight now can be assigned to the respective application.

In the next step, the individual cloud node workload is computed. By representing the cloud model as an arrangement of two interdependent row vectors, v-Mapper considers a cloud scenario where the number of tasks performed by the cloud is represented as a row vector $a = [a_1, a_2, a_3, \ldots, a_n]$ of a resource class c. Similarly, the resource occupation of individual tasks can be represented by another row vector $v = [v_1, v_2, v_3, \ldots, v_n]$. Each member of the row vector v represents the resource usage for its respective element in row vector a. By considering a case where the number of service requests accepted by the cloud environment does not exceed its aggregate compute nodes, the condition can be expressed mathematically as:

$$a.v = \sum_{i=1}^{n} a_c v_c \leq N \quad (7)$$

where N shows the number of aggregate compute nodes. Due to the varying allocation models and schemes, workload evaluation for cloud services is increasingly complex. However, applying probability to schemes with varying behavior can address these concerns. Therefore, by employing the recursive methodology approach presented in [27,28], we can calculate the probability p of an individual cloud node capacity q as:

$$p(q) = \sum_{a:av=n} \frac{\alpha_1^{a_1}}{\alpha_1!}, \ldots, \frac{\alpha_1^{a_n}}{\alpha_n!} n = 0, 1, 2, \ldots, N \quad (8)$$

The measured node occupancy probability can be input to the Fuzzy-AHP for fine tuning its decision structuring. Now, the submitted workload occupancy probability in the proposed cloud system can be evaluated. To ensure it upholds the workload occupancy of the entire cloud system, v-Mapper ensures that the resource allocation is made under the cloud's permissible resource threshold limits. The workload admission control algorithm (Algorithm 2) ensures that sufficient resources are available to entertain a request. If resources are scarce, a message about the unavailability of resources is prompted. This conditional approach used in v-Mapper makes it simple yet effective to implement. In this experimental study, the available resources of a cloud are considered as a set containing the cloud's physical resources, the computational resources, the memory and the bandwidth resources.

Algorithm 2. VM workload admission

```
1:  Input: Workload admission request
2:  Output: Workload admission decision
3:  Let res_ava denote the available resources for a VM
4:  Let res_req denote the requested resources by a VM
5:  Let queue denote the request queue
6:  while (!queue.isEmpty())
7:  {
8:          req = queue.firstRequestResource();
9:          if (res_req ≤ res_ava)
10:         {
11:         res_ava = res_ava − res_req
12:         queue.pop();
13:         }
14:         Else
15:         {
16:                 sendMessage("resources inadequate");
17:                 queue.pop();
18:         }
19: }
```

4.3. VM Scheduling Scheme

v-Mappers' scheduling policy (for admitted applications) involves locating resources that are subject to multiple global optimization constraints and that require searching of a large pool of resources. The proposed scheduler satisfies these requirements by using a fair approach to managing workloads. Our study considered three scenarios (Figure 4) to evaluate v-Mapper's scheduling policy.

Scenario 1: The VM hosting the applications is manually scheduled to receive batch processing of the incoming resource demands. The demands accepted by the VCPU of the VMM are synchronized before processing them.

Scenario 2: The scheduling strategy for real-time scheduling is set up in such a way that all the VCPUs contain the upcoming scheduling requests based on a load-sharing policy (LSP) [29], which optimally dispatches the incoming workloads according to the current availability of the VMs. The LSP facilitates the scheduling process by permanently engaging the minimal number of available VMs for the incoming workloads.

Scenario 3: We developed a load-sharing scheme that manages the VMs on the basis of the incoming workloads, enabling all the VCPUs to be scheduled and synchronized in response to the received workloads. This eases VM scheduling and improves the comprehensive understanding of the virtual infrastructure at both the physical and logical levels.

Theoretical analysis: v-Mapper uses a load-sharing concept to improve fairness. Usually, scheduling policies are governed by optimizing a generalized proportional fairness objective. Therefore, compared to other scheduling techniques, our approach enforces a more stringent fairness and load balance among workloads. Specifically, it assigns tags to each request when they arrive, and then dispatches requests in a non-descending order of tags. v-Mapper also delivers a statistically fair service even when the service rate fluctuates due to varying link capacity or varying CPU processing rates, etc.

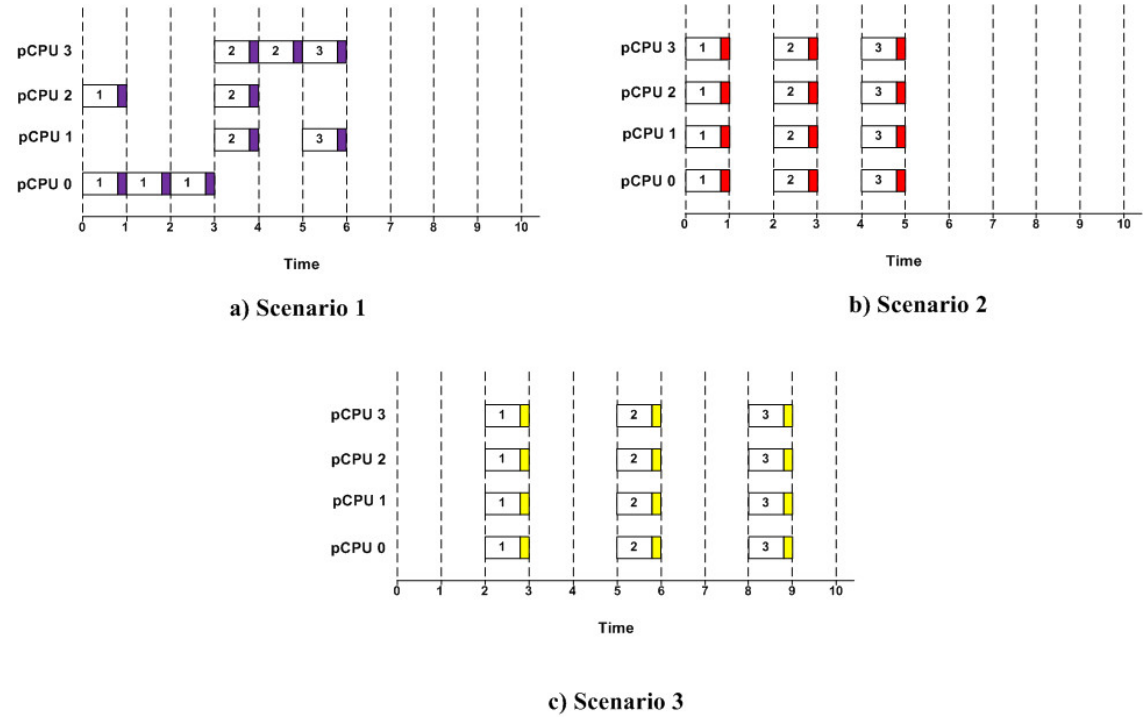

Figure 4. v-Mapper-Scheduling scenarios.

To demonstrate our load-sharing approach, let Ra be a scheduling request that carries a workload capacity Wa. This request is shared among r resources of a cloud's total available shared resources SR. The shared fair load SFL of an admission-control request Ra for each shared cloud resource SR is then defined as:

$$W_a^{SFL} = {}^{W_a}/_{r} \tag{9}$$

The shared fair load SFL of the shared resource SR for the scheduling request query Qreq can be defined as:

$$S_R^{SFL} = \sum_{R_a \in Q_{req}} W_a^{SL} \tag{10}$$

The scheduler prioritizes and accepts resource-intensive applications requiring maximum load-share capacity because their processing requirements fit in most with the system capacity. We then assign a priority Prq to the individual resource demand (per requesting application), whereby we consider the priority as a workload admission control request: shared fair load ratio. For the described scenario, the workload admission control request's shared fair load ratio can be expressed as:

$$Q_R / S_R^{SFL} \tag{11}$$

We next define a priority function (line 1 in Algorithm 3) to ensure that the service requests (i) are treated fairly in the queue (memory buffer). We apply conditionality (line 2) for execution, where a request query can only be entertained once all the resources required for its execution are available. This also guarantees that the total resource requirements of the requesting application cannot exceed the system threshold capacity (Rt) (see line 2). We then provide individual applications with their respective priority value (Pqi) (line 3). This priority value is calculated by the priority function (Equation (11)). In the next step, we queue all these buffered requests (br) in descending order (lines 7–13). This is done to ensure that the requests with the smallest resource requests but higher priority value are entertained first. Finally, the sorted resource-demanding applications are forwarded for process execution (lines 15–16). v-Mapper's VM scheduling policy is presented in Algorithm 3.

Algorithm 3. VM scheduling policy

Input: qi: represent a new request i; Rqi: the (required) resources for a service request i; Rt: system threshold for the resource, Pqi: priority of request i; N: request number in a memory buffer br.
Output: scheduling decision

Let priority = $\frac{Q_R}{S_R^{SFL}}$
while Rqi $\leq$ Rt **do**
    Assign request q_i its calculated priority value
    Push q_i to br
end while
/* Sort buffer requests by priority in descending order */
for I $\leftarrow$ 0 to N **do**
    for j $\leftarrow$ 0 to N-2-i **do**
        if $Pq_j < Pq_{(j+1)}$ **do**
            $Pq_j \leftrightarrow Pq_{(j+1)}$
        end if
    end for
end for
/* Process all sorted VM requests */
for i $\leftarrow$ 0 to N **do**
    Process(q_i)
end for
exit

5. Performance Test Bed and Simulation Environment

Below, we present a detailed study on the simulation settings of our system. The datacenter architectures used in the simulation were Fat-Tree and B-Cube, while the resource distributions used in the experiments were drawn from an empirical distribution given in [30]. The study report here covers our designed baseline strategies, the simulation environment and the results (Section 6).

5.1. Baseline Strategies and Compared Scenarios

In this section, we report on the evaluation of our algorithms against various data center topologies. To provide benchmarks for our evaluations, we considered the following baseline strategies.

Displacement strategy (ds): The displacement strategy selects a path with minimum end-to-end node congestion. It is frequently used in VM clusters handling small data loads.

Blind placement (bp): In a blind placement strategy, the tenant VMs adopts the path with a minimum hop count.

Sequential placement (sp): Here, the tenant VMs are placed sequentially in a server queue, where the VMs are ordered on the basis of their increasing machine-id order (lowest to highest order). This strategy attempts to balance the load resilience of applications by improving host subscription.

Stochastic (Random) placement (stp): Also known as the random placement strategy, in SP, the VMs are distributed in stochastic order. The stochastic placement strategy randomly picks available host machines from within a specified availability zone for a job.

5.2. Simulation Set-Up and Settings

We considered two types of topologies: physical topology and logical topology. To ensure optimal switching functions and traffic control, the physical cloud infrastructure used in our testing scenario consisted of eight hosts connected over seven SDN-enabled switches. The framework's prototype was implemented by using an API that uses SDN-enabled EPC Gateway to correspond with SDN-enabled switches. The simulations were performed through OpenDayLight controller and Mininet v.2.2.0 on an AMD Opteron™ 6300 Series Platform with 16-core x86 processor.

For the logical topology, we constructed a workload emulating the DC traffic. We also ran our experiments using a real workload from large computing clusters at Wuhan University. The workload contained 6 days of cluster machine activity and included details like job number, arrival, waiting and service time, CPU and memory usage traces. To avoid resource contention issues, the workload was experimented in controlled environment and workload was submitted in batches. During test runs it was considered that all links are active. We compared five algorithms i.e., vmap (v-Mapper), sp-bp, sp-ds, stp-bp and stp-ds on two datacenter topologies: Fat-Tree and B-Cube. Fat-tree is a scalable design for network architecture. The unique feature of a fat-tree data center architecture is that for any switch, the number of links going down to its siblings is equal to the number of links going up to its parent in the upper level. Therefore, the links get "fatter" towards the top of the tree. The switch in the root of the tree has the most links compared to any other switch below it. B-Cube [31], on the other hand, is server-centric network architecture in which servers with multiple network ports connect to multiple layers of mini-switches, as shown in Figure 5. By employing a range of topology settings, traffic and data applications, we ensured maximum core switch use to estimate the network resources use.

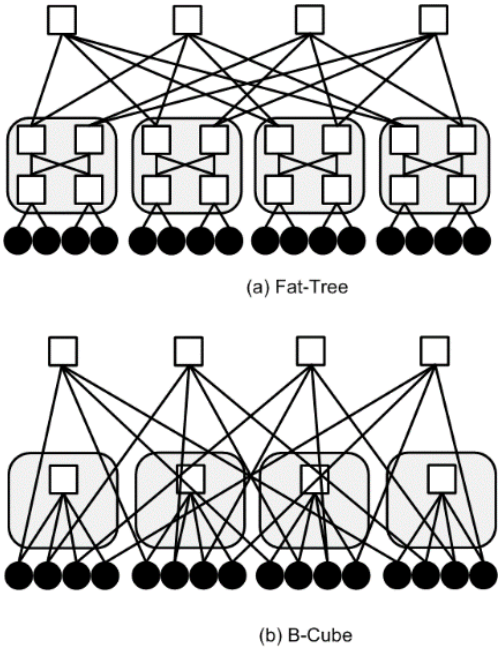

Figure 5. Topologies used in the performance evaluation.

6. Performance Results and Analysis

In this section, we present our findings to verify the effectiveness of the v-Mapper system.

6.1. Impact on Data Center Topologies

Our proposed strategy takes into account the measured (real-time) communication dependencies among VMs and the underlying data center network topology as well as the capacity limits of the physical servers of the data center. Its major goal is to minimize the data center network traffic while satisfying all the server-side constraints. We assign every link in every topology the same capacity. To ensure uniformity, a synthesized traffic pattern is adopted. A detailed literature review on data traffic patterns and its challenges are presented in [32–44] The VM resources are uniformly distributed and occur between range values of 0.2 and 0.8. From the results in Figure 6, vmap shows less congestion in the maximum core switch for the Fat-Tree and B-Cube topologies.

Our technique minimizes the path length between two intercommunicating VMs. This leads to an efficient and better usage of the link and switches resources. It also results in an increase in

the blocking ratio of groups of VMs with a high intensive traffic rate. Therefore, the performance improvement of vmap is more significant in topologies with better connectivity and path diversity.

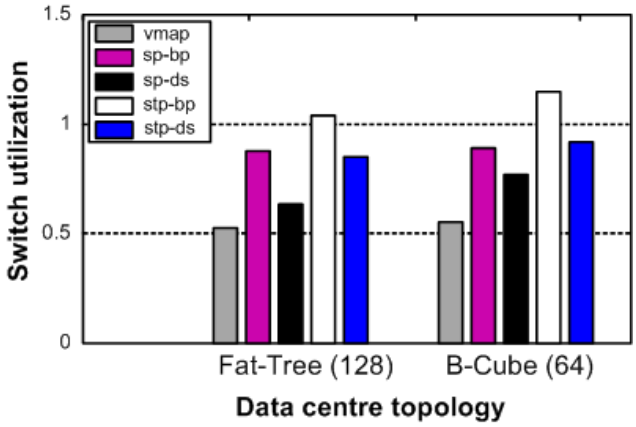

Figure 6. Performance evaluation of the algorithms with the different data center topologies.

6.2. Performance Cost

Performance can be assessed under numerous aspects. Here, we investigated scenarios where the hardware resources could be fully used. To evaluate the performance cost, we evaluated the performance of all the algorithms using a real workload obtained from our computing clusters. The details of 2500 jobs were extracted to evaluate their arrival times and flow duration statistics. These details were then catered for the Fat-Tree and B-Cube topologies. Figures 7 and 8 illustrate the detailed performance cost for all the compared algorithms with static and varying data traffic between VMs, respectively. From the results, it is clearly observed that as we increase the intensity of traffic between VMs, the overall performance cost increases. This increase results in congestion among communicating VMs and the slowing down of the networks' overall performance. Due to this, the performance of stp-bp declines. This can be understood as the stp-bp strategy is dependent on hop counts; therefore, if the traffic intensity among VMs is high, it adversely affects the algorithm's performance. On the other hand, vmap works independent of these constraints and has therefore the minimum cost among all the tested algorithms in the experimental set-up.

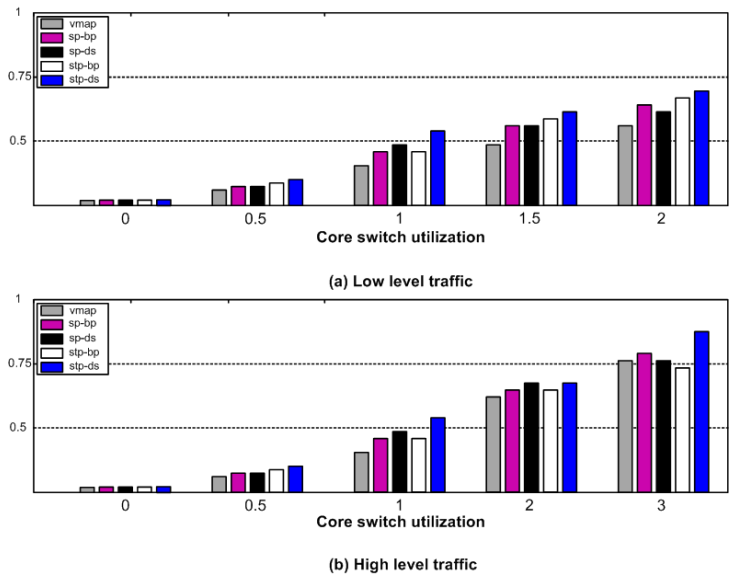

Figure 7. Performance evaluation of the algorithms using the Fat-Tree (128 node) data center topology.

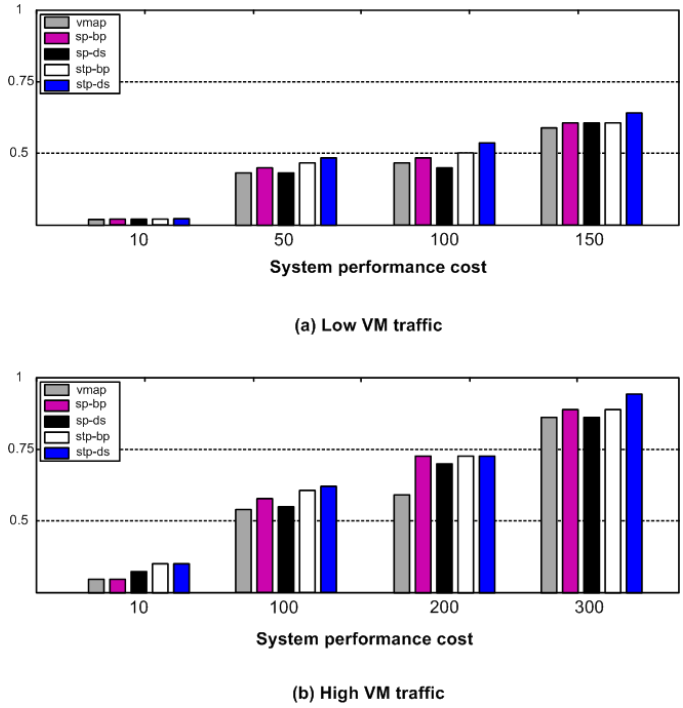

Figure 8. Performance evaluation of the algorithms by varying the traffic intensity between VMs.

6.3. VM Resource Occupancy

The distribution of VM image data should be based on the unit of chunks, instead of on entire VM image files. Herein, we first studied the distribution of the number of instances created from individual VMs. The VM size distribution is the amount of resources required by each VM instance to carry out discrete events. This categorization provides the basis for understanding how VM images are constructed and how similar they are. Different data center applications have their own patterns of resource consumption, e.g., CPU-intensive applications or memory-intensive applications, etc. To represent a clean and simple representation of a proposed scheme, our study defines only one type of resource consumption pattern. The evaluations were performed on 125-host B-Cube architecture with a synthesized traffic pattern. The application requests were categorized in four major categories i.e., uniform small, uniform large, random and bimodel demand. Further detail of the application request ranges employed in our experiments is given in Table 2. The results in Figure 9 demonstrate that vmap outperformed the other algorithms in the uniform small and random schemes; whereas in the uniform large and bimodal demand schemes, its efficiency showed a 30 percent improvement compared to the other algorithms. This makes vmap (v-Mapper) a simple, feasible, yet practical VM management and resource consolidation solution.

Table 2. Classification of VM resources.

Size	Request Range
Uniform small	30 percent
Uniform large	100 percent
Random	0 to 100 percent
Bimodel demand	Normal distributions N (0:3;0:1) and N (0:7;0:1) with equal probability

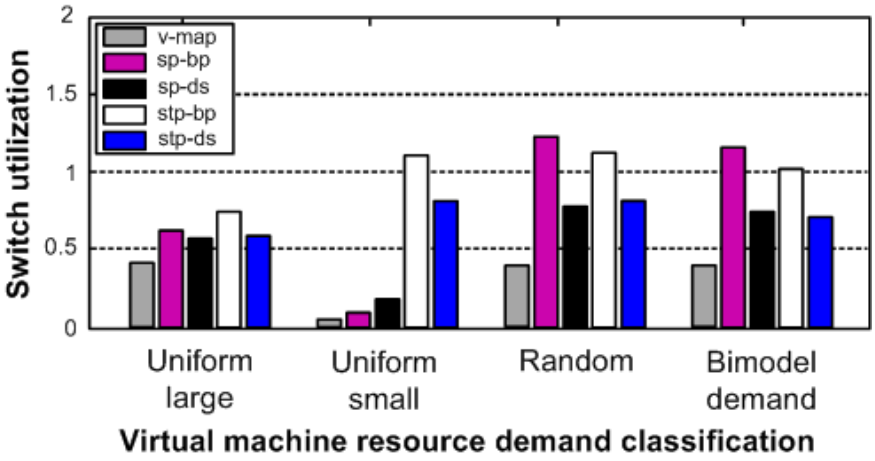

Figure 9. Switch resource use for the compared resource allocation schemes.

7. Conclusions and Future Work

With the prevalence of virtualization and multi-core processing architectures, hosting VMs and their applications is becoming a challenging task in cloud data centers. In this paper, we optimized VM placement through a graph theoretic approach under controlled parameters. The representation of the VM placement problem presented in this paper and the use of matrices (from linear algebra) result from the fact that cloud systems can be represented as a network of nodes and edges forming a graph. Next, we proposed an algorithm that solves the scheduling of applications using an online solution for a dynamic environment with changing network traffic. By leveraging and expanding capacity approximation techniques, we propose v-Mapper, which is an efficient solution that restricts the synthesized data center traffic traces under a spectrum of workloads. The proposed system was tested using three baseline strategies. v-Mapper strategy takes into account the real-time communication dependencies among VMs and the underlying data center network topology along with the capacity limits of the physical servers of the data center. Results demonstrate that v-Mapper minimizes the path length between two intercommunicating VMs and outperforms other compared techniques in terms of VM resource usage and occupancy. In proposed study, we ignored the limitations of VM overheads. In future, we plan to exploit the effects of topology changes for our framework. We also plan to investigate a best-fit approximation solution for reducing the VM overheads.

Author Contributions: A.A.A. and H.J. designed the study. H.J. designed experiments. A.A.A. performed experiments. A.A.A. and H.J. analyzed the manuscript.

Acknowledgments: The authors want to thank anonymous reviewers for their valuable suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Abbasi, A.A.; Jin, H.; Wu, S. A Software-Defined Cloud Resource Management Framework. In Proceedings of the Asia-Pacific Services Computing Conference, Bangkok, Thailand, 7–9 December 2015.
2. Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A.D.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, A.; Stoica, I.; et al. A view of cloud computing. *Commun. ACM* **2010**, *53*, 50–58. [CrossRef]
3. Wu, S.; Zhou, L.; Sun, H.; Jin, H.; Shi, X. Poris: A Scheduler for Parallel Soft Real-Time Applications in Virtualized Environments. *IEEE Trans. Parallel Distrib. Syst.* **2016**, *27*, 841–854. [CrossRef]
4. Wu, S.; Chen, H.; Di, S.; Zhou, B.; Xie, Z.; Jin, H.; Shi, X. Synchronization-Aware Scheduling for Virtual Clusters in Cloud. *IEEE Trans. Parallel Distrib. Syst.* **2015**, *26*, 2890–2902. [CrossRef]
5. Greenberg, A.; Hamilton, J.; Maltz, D.A.; Patel, P. The cost of a cloud: Research problems in data center networks. *Comput. Commun. Rev.* **2008**, *39*, 68–73. [CrossRef]

6. Liao, X.; Jin, H.; Liu, Y.; Ni, L.M.; Deng, D. AnySee: Peer-to-Peer Live Streaming. In Proceedings of the IEEE International Conference on Computer Communications, Barcelona, Spain, 23–29 April 2006; pp. 1–10.
7. Kim, H.; Feamster, N. Improving network management with software defined networking. *IEEE Commun. Mag.* **2013**, *51*, 114–119. [CrossRef]
8. Monsanto, C.; Reich, J.; Foster, N.; Rexford, J.; Walker, D. Composing software defined networks. In Proceedings of the USENIX Symposium on Networked Systems Design and Implementation, Lombard, IL, USA, 2–5April 2013; pp. 1–13.
9. Jin, H. When Data Grows Big. *Computer* **2014**, *12*, 8. [CrossRef]
10. Wang, M.; Meng, X.; Zhang, L. Consolidating virtual machines with dynamic bandwidth demand in data centers. In Proceedings of the IEEE International Conference on Computer Communications, Shanghai, China, 10–15 April 2011; pp. 71–75.
11. Piao, J.T.; Yan, J. A network-aware virtual machine placement and migration approach in cloud computing. In Proceedings of the International Conference on Grid and Cooperative Computing, Nanjing, China, 1–5 November 2010; pp. 87–92.
12. Meng, X.; Pappas, V.; Zhang, L. Improving the scalability of data center networks with traffic-aware virtual machine placement. In Proceedings of the IEEE International Conference on Computer Communications, San Diego, CA, USA, 15–19 March 2010; pp. 1–9.
13. Wang, S.H.; Huang, P.P.W.; Wen, C.H.P.; Wang, L.C. EQVMP: Energy-efficient and QoS-aware virtual machine placement for software defined datacenter networks. In Proceedings of the IEEE International Conference on Information Networking, Phuket, Thailand, 10–12 February 2014; pp. 220–225.
14. Bobroff, N.; Kochut, A.; Beaty, K. Dynamic placement of virtual machines for managing SLA violations. In Proceedings of the 10th IFIP/IEEE International Symposium on Integrated Network Management, Munich, Germany, 21–25 May 2007; pp. 119–128.
15. Gong, Z.; Gu, X. Pac: Pattern-driven application consolidation for efficient cloud computing. In Proceedings of the IEEE International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems, Miami, FL, USA, 17–19 August 2010; pp. 24–33.
16. Calcavecchia, N.M.; Biran, O.; Hadad, E.; Moatti, Y. VM placement strategies for cloud scenarios. In Proceedings of the IEEE International Conference on Cloud Computing, Honolulu, HI, USA, 24–29 June 2012; pp. 852–859.
17. Katta, N.P.; Rexford, J.; Walker, D. Incremental consistent updates. In Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, Hongkong, China, 16 August 2013; pp. 49–54.
18. Foster, N.; Guha, A.; Reitblatt, M.; Story, A.; Freedman, M.J.; Katta, N.P.; Mosanto, C.; Reich, J.; Rexford, J.; Schlesinger, C.; et al. Languages for software-defined networks. *IEEE Commun. Mag.* **2013**, *51*, 128–134. [CrossRef]
19. Banikazemi, M.; Olshefski, D.; Shaikh, A.; Tracey, J.; Wang, G. Meridian: An SDN platform for cloud network services. *IEEE Commun. Mag.* **2013**, *51*, 120–127. [CrossRef]
20. Sherry, J.; Hasan, S.; Scott, C.; Krishnamurthy, A.; Ratnasamy, S.; Sekar, V. Making middleboxes someone else's problem: Network processing as a cloud service. *Comput. Commun. Rev.* **2012**, *42*, 13–24. [CrossRef]
21. Jin, H.; Abbasi, A.A.; Song, W. Pathfinder: Application-aware distributed path computation in clouds. *Int. J. Parallel Prog.* **2017**, *45*, 1273–1284. [CrossRef]
22. Binz, T.; Breiter, G.; Leyman, F.; Spatzier, T. Portable cloud services using tosca. *IEEE Internet Comput.* **2012**, *16*, 80–85. [CrossRef]
23. Mahindru, R.; Sarkar, R.; Viswanathan, M. Software defined unified monitoring and management of clouds. *IBM J. Res. Dev.* **2014**, *58*, 1–12. [CrossRef]
24. Chang, D.; Xu, G.; Hu, L.; Yang, K. A network-aware virtual machine placement algorithm in mobile cloud computing environment. In Proceedings of the IEEE Wireless Communications and Networking Conference Workshops, Shanghai, China, 7–10 April 2013; pp. 117–122.
25. Zadeh, L.A. Fuzzy sets. *Inf. Control* **1965**, *8*, 338–353. [CrossRef]
26. Saaty, T.L. What is the analytic hierarchy process? In *Mathematical Models for Decision Support*; Springer: Berlin, Germany, 1988; pp. 109–121.
27. Bonald, T.; Virtamo, J. A recursive formula for multirate systems with elastic traffic. *IEEE Commun. Lett.* **2005**, *9*, 753–755. [CrossRef]

28. Kaufman, J.S.; Rege, K.M. Blocking in a shared resource environment with batched Poisson arrival processes. *Perform. Eval.* **1996**, *24*, 249–263. [CrossRef]
29. Cherkasova, L.; Gupta, D.; Vahdat, A. Comparison of the three CPU schedulers in Xen. *Perform. Eval. Rev.* **2007**, *35*, 42–51. [CrossRef]
30. Jiang, J.W.; Lan, T.; Ha, S.; Chen, M.; Chiang, M. Joint VM placement and routing for data center traffic engineering. In Proceedings of the IEEE International Conference on Computer Communications, Orlando, FL, USA, 25–30 March 2012; pp. 2876–2880.
31. Guo, C.; Lu, G.; Li, D.; Wu, H.; Zhang, X.; Shi, Y.; Tian, C.; Zhang, Y.; Wu, S. BCube: A high performance, server-centric network architecture for modular data centers. *Comput. Commun. Rev.* **2009**, *39*, 63–74. [CrossRef]
32. Shojafar, M.; Cordeschi, N.; Baccarelli, E. Energy-efficient adaptive resource management for real-time vehicular cloud services. *IEEE Trans. Cloud Comput.* **2016**. [CrossRef]
33. Shojafar, M.; Canali, C.; Lancelotti, R.; Abawajy, J. Adaptive computing-plus-communication optimization framework for multimedia processing in cloud systems. *IEEE Trans. Cloud Comput.* **2016**. [CrossRef]
34. Canali, C.; Chiaraviglio, L.; Lancellotti, R.; Shojafar, M. Joint Minimization of the Energy Costs from Computing, Data Transmission, and Migrations in Cloud Data Centers. *IEEE Trans. Green Commun. Netw.* **2018**, *2*, 580–595. [CrossRef]
35. Chiaraviglio, L.; D'Andregiovanni, F.; Lancellotti, R.; Shojafar, M.; Melazzi, N.B.; Canali, C. An Approach to Balance Maintenance Costs and Electricity Consumption in Cloud Data Centers. *IEEE Trans. Sustain. Comput.* **2018**. [CrossRef]
36. Rezaei, R.; Chiew, T.K.; Lee, S.P.; Shams, A.Z. A semantic interoperability framework for software as a service systems in cloud computing environments. *Exp. Syst. Appl.* **2014**, *41*, 5751–5770. [CrossRef]
37. Shiau, W.L.; Chau, P.Y.K. Understanding behavioral intention to use a cloud computing classroom: A multiple model comparison approach. *Inf. Manag.* **2016**, *53*, 355–365. [CrossRef]
38. Dillon, T.; Wu, C.; Chang, E. Cloud computing: Issues and challenges. In Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA'10), Perth, Australia, 20–23 April 2010; pp. 27–33.
39. Fan, M.; Kumar, S.; Whinston, A.B. Short-term and long term competition between providers of shrink-wrap software and software as a service. *Eur. J. Oper. Res.* **2009**, *196*, 661–671. [CrossRef]
40. Palos-Sanchez, P.R.; Arenas-Marquez, F.J.; Aguayo-Camacho, M. Cloud Computing (SaaS) adoption as a strategic technology: Results of an empirical study. *Mob. Inf. Syst.* **2017**, *2017*, 2536040. [CrossRef]
41. Sultan, N.A. Reaching for the "cloud": How SMEs can manage. *Int. J. Inf. Manag.* **2011**, *31*, 272–278. [CrossRef]
42. Palos-Sanchez, P.R. Drivers and Barriers of the Cloud Computing in SMEs: The Position of the European Union. *Harvard Deusto Bus. Res.* **2017**, *6*, 116–132. [CrossRef]
43. Malik, S.U.R.; Khan, S.U.; Ewenet, S.J. Performance analysis of data intensive cloud systems based on data management and replication: A survey. *Distrib. Parallel Databases* **2016**, *34*, 179–215. [CrossRef]
44. Haag, S.; Eckhardt, A. Organizational cloud service adoption: A scientometric and content-based literature analysis. *J. Bus. Econ.* **2014**, *84*, 407–440. [CrossRef]