

Network and Server Resource Management Strategies for Data Centre Infrastructures: A Survey

Fung Po Tso^{a,*}, Simon Jouet^b, Dimitrios P. Pezaros^b

^a*Department of Computer Science, Liverpool John Moores University, L3 3AF, UK*

^b*School of Computing Science, University of Glasgow, G12 8RZ, UK*

Abstract

The advent of virtualisation and the increasing demand for outsourced, elastic compute charged on a pay-as-you-use basis has stimulated the development of large-scale Cloud Data Centres (DCs) housing tens of thousands of computer clusters. Of the significant capital outlay required for building and operating such infrastructures, server and network equipment account for 45% and 15% of the total cost, respectively, making resource utilisation efficiency paramount in order to increase the operators' Return-on-Investment (RoI).

In this paper, we present an extensive survey on the management of server and network resources over virtualised Cloud DC infrastructures, highlighting key concepts and results, and critically discussing their limitations and implications for future research opportunities. We highlight the need for and benefits of adaptive resource provisioning that alleviates reliance on static utilisation prediction models and exploits direct measurement of resource utilisation on servers and network nodes. Coupling such distributed measurement with logically-centralised Software Defined Networking (SDN) principles, we subsequently discuss the challenges and opportunities for converged resource management over converged ICT environments, through unifying control loops to globally orchestrate adaptive and load-sensitive resource provisioning.

Keywords: Cloud Data Centre, Virtualisation Management, Network

*Corresponding author

Email address: p.tso@ljmu.ac.uk (Fung Po Tso)

1. Introduction

Cloud computing is an important IT paradigm where enterprises outsource ICT infrastructure and resources based on a *pay-as-you-use* service model. This model relieves enterprises from significant capital expenditure (CAPEX) costs for purchasing and maintaining in-house permanent hardware and software assets. Instead, they use operating expense budgets (OPEX) to fund their ICT infrastructure and eliminate maintenance expenses, allowing them to focus on core business innovation. One of the most immediate benefits of using Cloud services is the ability to increase infrastructural capacity swiftly and at lower costs, therefore being able to adapt to changes in the market without complex procurement processes, and respond flexibly to unexpected demand. Recent years have witnessed a significant growth in the adoption of Cloud Computing. The public Cloud computing market has expanded by 14% in 2015 to total US\$175 billion, according to Gartner Inc. [1], whilst total spending worldwide is anticipated to continue flourishing at a Compound Annual Growth Rate (CAGR) of 17.7% until 2016 [2].

Underpinning Cloud Computing are virtualised infrastructures hosted over Data Centres (DCs) which are in turn maintained and managed at scale by national or global operators, such as Amazon, Rackspace, Microsoft, and Google. These implement different variations of the *-as-a-Service (*aaS) paradigm, including Infrastructure-as-a-Service (IaaS, e.g., Amazon's EC2 and Google's Compute Engine), Platform-as-a-Service (PaaS, e.g. Microsoft's Azure and Rackspace's Cloud Sites) and Software-as-a-Service (SaaS, e.g., Facebook and Google Docs).

It was anecdotally reported that the number of servers owned by some major Cloud service providers and operators could be more than a million [3]. They are hosted in Cloud DCs, each typically housing tens of thousands of servers [4]. In order to be sustainable, the significant capital outlay required for building

a DC makes maximisation of Return on Investment (RoI) crucial, which in turn necessitates efficient and adaptive resource usage of the virtualised physical infrastructure [5][6][7]. With the advent of virtualisation and multi-tenancy, computing resources are shared amongst multiple tenants, preventing hard resource commitment and therefore servers from being idle. However, this soft resource allocation results in significant load fluctuation in short timescales due to the ebb and flow of user demand.

DC infrastructures are therefore vulnerable to performance degradation from factors such as network congestion and contention on shared resources. Managing such dynamism in short timescales is particularly challenging. Many Cloud applications such as, Hadoop running over Cloud DCs exploit a fine-grained hierarchical task decomposition into stages, each of which can involve multiple instances running in parallel on different physical hosts and communicating between them. In some intermediate stages, computation results yielded from subtasks are gathered on to fewer number of servers to produce input for subsequent and final compute stages. In this partition/aggregate work pattern, it is essential for all subtasks to complete in time in order for a job to complete since any failed subtask will have to be re-executed and keep others “waiting” for it to complete. As a result, the completion time of each single subtask ultimately determines the overall job completion time. This is wasteful both for CPU cycles and network bandwidth, and can have a knock-on effect on the response time of different services and different tenants.

Consequently, DC resource management has become a complex problem due to the inability to gather accurate infrastructure-wide resource usage information in short timescales and in turn to forecast resource availability. Recent research has revealed that DC workload patterns at coarse time-scales (i.e., hours) exhibit weekend/weekday variations [8], but at finer-grained timescales, the workload patterns are bursty and unpredictable [6][9][10]. The measurement results indicate that in order to adapt to transient load fluctuation, a fine-grained temporal and spatial approach is needed. For fine temporal granularity, control loops are needed to obtain levels of resource utilisation in short

60 timescales for better characterising workload patterns. For spatial granularity, individual flows size, server availability, network link utilisation, etc., need to be measured and used as additional input to resource provisioning algorithms [10].

Currently, to cope with performance variability and unpredictability, DCs are engineered to tolerate a certain degree of demand fluctuation by over-provisioning, or by holding certain resources as a reserve [11][12]. However, 65 over-provisioning within Cloud DCs is expensive [4]. Alternatively, adaptive provisioning policies can be implemented to ensure that Cloud providers adhere to their Service Level Agreements (SLAs) while maximising the utilisation of the underlying infrastructure.

70 Cloud resource management requires complex instrumentation mechanisms and algorithms for multi-objective optimisation to measure and account for e.g., server, network, and power usage efficiency. In this paper, we provide a critical survey of resource management strategies for virtualised Data Centre infrastructures. We focus on two key infrastructural aspects: the servers and 75 the underlying network. These two pillars not only represent the most costly infrastructural elements, up to 60% of the cost of a data centre [4], that need to be managed and provisioned in an efficient and effective manner, they also adequately capture the level of granularity of resource contention and multiplexing over Cloud DCs [13]. In virtualised DCs, virtual Machines (VMs) are the 80 fundamental entities used by users over both public and private infrastructure Clouds while traffic is multiplexed and controlled at the level of individual flows over the DC network. We discuss management strategies for the static and dynamic allocation of virtual resources over physical servers to improve response times, power and energy consumption, and network bandwidth utilisation. We 85 present the network-wide characteristics of typical DC workloads, and we review the most prominent work on traffic engineering strategies to achieve different network-wide objectives. In addition, we discuss developments on traffic flow admission and congestion control for Cloud DCs that primarily seek to harness the underlying redundancy in network bandwidth to maximise intra-DC pair-wise application throughput. In each of these areas, we highlight the limitations 90

of the state-of-the-art and we discuss efforts towards more adaptive and more dynamic, closed-loop resource management and control.

The remainder of this paper is structured as follows: Section 2 presents the dominant DC network architectures and management topologies used to leverage network and server resource redundancy and enable horizontal (rather than vertical) infrastructure expansion. We then critically discuss the most important and influential developments on server, network, and flow control management over Cloud DCs, in sections 3, 4 and 5, respectively. Within each category, we present the main optimisation objectives and main techniques for achieving them. We identify areas for future development and open research issues that are yet to be addressed. In Section 6, we raise the issue of the current disjoint management and control of diverse physical and virtual resources in the DC, and we discuss the inefficiencies this lack of synergy between control mechanisms can lead to. We then describe research challenges and opportunities for converged control and resource management for virtualised Cloud DCs. Finally, Section 7 concludes the paper.

2. Data Centre Topologies

In this section, we provide a critical review of the dominant Cloud Data Centre (DC) network architectures, outlining their operational characteristics, limitations and expansion strategies.

2.1. Conventional DC Architecture

Conventional DC architectures are built on tree-like hierarchy with high-density, high-cost hardware [14], as depicted in Fig. 1a. The network is a tree containing a layer of servers arranged in racks at the bottom. Each server rack typically hosts 20-40 servers connected to a Top of Rack (ToR) switch with a 1Gb/s link. Each ToR switch connects to two aggregation switches for redundancy. For the same reason, aggregation switches connect to core switches/routers that manage traffic in and out of the DC. The hierarchical

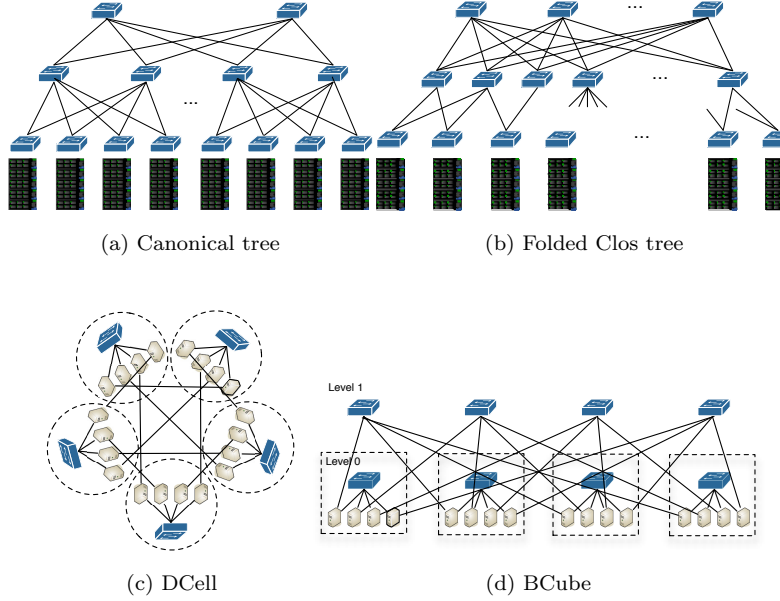


Figure 1: Switch-centric ((a) & (b)) and server-centric ((c) & (d)) Cloud data centre topologies.

configuration of the network topology means that traffic destined to servers
 120 in different racks must go through the aggregation or the core switches of the
 network. Therefore, aggregation switches usually have larger buffers as well
 as higher throughput and port density, and are significantly more expensive
 than ToR switches. To make the network fabric cost-effective, higher layer links
 are typically oversubscribed by factors of 10:1 to 80:1, limiting the bandwidth
 125 between servers in different branches [4]. Links in the same rack are not over-
 subscribed and thus collocated servers can operate at full link rate. Cross-rack
 communication is routed through the higher layers of the topology and therefore,
 in case of persistent and high-load communication between racks, the aggrega-
 tion and core switches can become congested and result in high latency and
 130 packet loss. To increase capacity, network operators must resolve to vertical ex-
 pansion, in which operators replace overloaded switches with higher-cost, higher
 capacity ones [15].

2.2. Clos/Fat-tree Architecture

Modern data centre architectures [16][17][6] have been proposed to reduce or even remove oversubscription altogether. Representative work, such as e.g., Clos-Tree [6] and Fat-Tree [16], promote horizontal rather than the traditional vertical expansion. Instead of replacing higher-layer costly switches, network operators can add inexpensive commodity switches to expand their network horizontally using a fat-tree topology (Clos topology for VL2). Dense interconnect in these new fabrics provides a larger number of redundant paths between any given source and destination edge switch (i.e. rich equal cost path redundancy in contrast to only 2 equal cost path in the conventional DC architecture) meaning that oversubscription of the higher layer links can be significantly mitigated. In Clos-tree topology, as shown in Fig. 1b, links between the core layer switches and the aggregation layer switches form a complete bipartite graph [6]. However, ToRs only connect to two aggregation switches as in the conventional tree architecture. The limiting factor for the size of a fat-tree fabric is determined by the number of ports on switches. Fat-tree uniformly uses k -port commodity switches at all layers. All switches at the edge and aggregation layers are clustered into k pods, each containing k switches (i.e. $k/2$ switches at each layer). In other words, edge layer switches have $k/2$ remaining ports to connect to $k/2$ hosts. Similarly aggregation layer switches use the remaining $k/2$ ports for connecting $(k/2)^2$ k -port core switches. Eventually, a k -ary fat-tree can support up to $k^3/4$ hosts.

Clos/Fat-tree architectures have seen an increasing popularity in modern data centres to achieve high performance and resiliency through their ability to provide better scalability and path diversity than conventional DC topologies [18][19]. However, these architectures require homogeneous switches, and large numbers of links. Upgrading to these architectures in a legacy DC usually requires replacing most existing switches and cables. Such radical upgrades are typically prohibitively expensive and time-consuming [20].

2.3. Server-centric Architecture

In server-centric DC architectures, servers are both end-hosts and relaying nodes for multi-hop communications. The most representative fabrics are BCube [21] and DCell [22]. Both BCube and DCell come with custom routing protocols to take advantage of topological properties [23]. As shown in Fig. 1c, in DCell, a server is connected to a number of servers in other cells and to a switch in its own cell. According to [22], a high-level DCell is constructed from low-level DCells (DCell_k , $k \geq 0$) in a recursive manner. DCell_0 , as shown in Fig. 1c, is the building block to construct larger DCells. It has n servers and a mini-switch ($n = 4$ for DCell_0 Fig. 1c) and all servers in DCell_0 are connected to the mini-switch. And then level-1 DCell_1 is constructed using $n + 1$ DCell_0 s. In DCell_1 , each DCell_0 is connected to all the other DCell_0 s with one link. And this procedure is repeated to create higher level DCells.

In comparison, a BCube_0 is n servers connected to an n -port switch. A BCube_1 is comprised of n BCube_0 's and n n -port switches [21]. In BCube, as illustrated in Fig. 1d, two servers are neighbours if they connect to the same switch. BCube names a server in a BCube_k using an address array $a_k a_{k-1} \dots a_0$ ($a_i \in [0, n - 1]$, $i \in [0, k]$). Two servers are neighbours if and only if their address arrays differ by a single digit. That is, two neighbouring servers that connect to the same level i switch are different at the i th digit [21].

A prominent competing advantage of server-centric architecture is the manageability. Since the entire DC fabric is built from servers and a minimal set of network switches, only a single team of engineers is required to maintain and manage the whole architecture. In contrast, multiple (internal and external) professional teams are needed for managing various switches, that are produced by different manufacturers, in switch-centric fabrics. Also, in a server-centric architecture, intelligence can be placed on servers for implementing in-network services such as traffic aggregation, caching as well as deep-packet inspection etc. However, a server-centric architecture is fundamentally different from traditional network designs and has been seen as an untrusted and complex to update option. In order to promote server-centric architectures, they should

offer more significant competing advantages including remarkable reduction in overall deployment cost, improvement in security and resilience. [24].

195 2.4. Management Topology

DC network nodes exchange considerable management data in order to configure and maintain the network-wide topology, and consequently, management data intensifies as the topology becomes denser. [25] reports that management traffic can account to approximately 5–10% of the bandwidth during normal
 200 operating conditions. With the recent advent of Software Defined Networking (SDN) [26], a paradigm that logically centralises and separates the network control from the data plane, the management network is tightly coupled with the control plane as control decisions must be transmitted between the switches and a central controller. The requirements of a management network are different from those of the data-carrying network: management traffic is sparse
 205 and maintaining high throughput is not critical, however, it is latency-sensitive and failures can be critical to the production network behaviour. Three different types of management networks have been covered in the literature, the simplest is to manage the network *in-band* (IB). In this configuration, both
 210 management and production traffic share the same network. This allows management to be cost-effective but in case of over-utilised production networks, the management network is also hindered. It is possible to mitigate hindrance to management traffic from production traffic through Quality of Service (QoS) enforcement to prioritise management traffic over data traffic [27]. The other
 215 approach is to have a logical or physical *Out-of-Band* (OOB) network. In a logical OOB network, the core of the network is still shared between management and production traffic, but logical isolation is achieved using VLANs or dedicated OpenFlow forwarding rules [28]. OpenFlow is a communication protocol and API providing access to the forwarding plane of network devices; it is
 220 the most widely deployed implementation of the SDN paradigm. In this case, each switch must have a dedicated port for management, increasing the cost as production traffic have less dedicated ports but limiting possible interference

between production and management traffic. However, such setup is still vulnerable to device failure or misconfiguration. Finally, a physical OOB network
 225 can be used in environments where the management network operation is critical, such as a SDN environment without graceful fallback to learning switches or other distributed mechanisms, when the controller is unreachable. In such environments, a different physical network is dedicated to management operations [29][25].

230 **3. Server Resource Management**

The cost of servers in data centres can account up to 45% of running cost per year [4]. It is apparent that achieving high server utilisation is of paramount importance in order to increase Return-on-Investment (RoI). However, server utilisation in DCs can be as low as 10% [30] due to over-provisioning as a result
 235 of the desire to provision for peak demand [31].

Achieving high server utilisation is challenging. First, it is difficult for DC operators and customers to plan in advance for “diurnal usage patterns, unpredictable spikes in user and traffic demand, and evolving workloads” [9]. Second, it is very expensive, if not impossible, for both of providers and consumers (who have little control and choice [32]) to configure individual servers
 240 so that fine-grained resources, such as, e.g., CPU, memory, storage and network, perfectly match temporal application requirements due to the heterogeneity of servers (i.e., servers have different CPU, RAM and other resource capacities) and the complexity associated with calculating individual resource requirements for different services [33]. Third, increasing server utilisation by
 245 scheduling multiple services on one physical host can cause severe performance degradation due to resource (e.g. CPU, Memory, Storage and I/O peripheral) contention [34][34][35]. Last, but not least, Cloud DC operators and service providers often need to meet strict QoS guarantees through Service Level Agreements (SLA). Meeting SLAs is crucial, since it gives confidence to customers to
 250 move their ICT infrastructure into the Cloud environments and heavy penalties

are paid by the provider if the SLA is not met. Typically, in order to meet SLA requirements, resources are over-provisioned to meet worst-case demand [36].

Server consolidation is the activity of clustering or reassigning several virtual machines (VMs) running on under-utilised physical servers into fewer hosts, and is used in DCs to improve resource utilisation and reduce operational expenditure (OPEX). VM consolidation has been employed by DC operators to optimise diverse objectives, such as, e.g., server resource (CPU, RAM, net I/O) usage [37][38][39] and energy efficiency [40][41][42], or to meet SLA requirements which are often expressed as CPU or response time guarantees [36][43]. Most server management works take one resource as optimisation objective and treat other resource as constraints or jointly consider multiple resource and SLA constraints. Hence, for the ease of discussion, the research works are broadly categorised based on their main optimisation objectives in the following discussions.

In fact, some production software such as VMware vSphere Distributed Resource Scheduler (DRS) [44], Microsoft System Center Virtual Machine Manager (VMM) [45], and Citrix XenServer [46] offer VM consolidation as one of their major features.

3.1. Types of VM consolidation

Server consolidation can be broadly classified as static or dynamic. In static consolidation, or *initial placement*, consolidation algorithms take historical resource utilisation as input to predict future resource use trend based on which VMs are mapped to physical hosts [47][48]. Once initial static consolidation has taken place, VM assignments usually remain unchanged for extended periods of time (e.g., months or even years). It is also done off-line due to the high complexity of consolidation algorithms. Static consolidation is ideal for static workload as it can achieve optimality. On the contrary, dynamic allocation is implemented over short timescales in response to change in resource demand by leveraging the ability to do live migration of VMs.

Dynamic server consolidation is particularly useful for unpredictable workload in which prediction-based mechanisms fail to work. Dynamic consolidation is carried out periodically in shorter timescale than static one to adapt to changes of work demand [39][49][50][40][51][52].

285 3.2. Server Resource-aware Consolidation Schemes

When the users' demand changes, VMs can start competing for physical resources resulting in computation hotspots. *Sandpiper* [39] is a tool that detects and mitigates hotspots based on physical machine resources such as CPU, network and memory. In order to detect hotspot, *Sandpiper* implements a
 290 monitoring and profiling engine that collects CPU, network and memory usage statistics on VMs and a time-series prediction technique (which relies on the auto-regressive family of predictors) to predict the likelihood of hotspots. The monitoring can be either unobtrusive black-box monitoring which infers CPU, network and memory usage of each VM from external observation (at the host)
 295 or a more aggressive gray-box monitoring that explicitly puts a daemon inside each VM to monitor/measure resource consumed by individual VMs. For both monitoring approaches, techniques are employed to estimate the peak resource needs. Upon detection of hotspot, it is the migration manager's responsibility to carry out hotspot mitigation. Since optimally deciding *which* and *where*
 300 to migrate is a NP-hard multi-dimensional bin packing problem, the migration manager employs a heuristic to migrate VMs from overloaded servers to underloaded servers where migration overhead (i.e., the amount of data transferred) is minimised. The main drawback of *Sandpiper* is that it only reactively triggers migration upon detection of hotspot within the infrastructure and does not
 305 consider the migration overhead.

Entropy [53] achieves optimal VM configuration while also ensures that every VM has access to sufficient memory and allocated CPU share. *Entropy* has a sensor that periodically probes VMs' CPU usage and working status. Any changes will trigger a reconfiguration process via migration, which consists of
 310 virtual machine packing problem (VMPP) and virtual machine re-placement

problem (VMRP). Constraint programming is then employed to solve VMPP and VMRP problems. However, in order to reduce migration decision time, when an optimal solution cannot be computed within 1 minute, the computation is aborted and the current best result is used. Entropy takes migration overhead
 315 into consideration when making migration decisions, however, it assumes that the resource demand is known and static over time.

Similar to *Sandpiper*, *ReCon* [51] exploits servers' historical resource usage to discover applications that can be consolidated and recommends a dynamic consolidation plan that can be deployed in a multi-cluster data centre. The
 320 VM consolidation is formulated as an optimisation problem with multiple constraints. The cost function is defined as the running cost of a physical server, predominated by power consumption which is translated into CPU utilisation. Hence, the objective is to minimise the cost given a set of VMs and constraints. However, prediction-based scheme can be sub-optimal when resource require-
 325 ment is dynamic.

In contrast to optimising resources as complete units in aforementioned works, multi-resource schedulable unit (MRSU) [54] breaks CPU, memory, storage and network into small chunks to tackle resource-overallocation problem at fine granularity. MRSU firstly determine schedulable unit in each resource
 330 dimension and then compute the number of MRSUs needed for particular instances. MRSU allocation is a min-max problem and hence a weighted fair heuristic is proposed to solve the problem.

3.3. Energy-aware Consolidation Schemes

VirtualPower [55] is a pioneer work to look into server power management
 335 in virtualised environments. When a server is virtualised and shared among guest VMs, its hardware power management cannot function properly due to diverse and inconsistent virtual servers' activities unless all virtual servers agree on the same limitation, e.g. reducing memory bandwidth, concurrently. On the other hand, guest VMs see themselves as independent server and proactively
 340 try to manage 'their power states'. Instead of ignoring these built-in power

management policies as done by hypervisor, *VirtualPower* exploits the policies as effective hints of individual VM’s power state. Therefore, *VirtualPower* can provide a rich set of ‘soft’ VirtualPower Management (VPM) states to VMs and then use VMs’ state changes requests as inputs to manage power locally
345 on individual physical server and globally (that considers maximum power consumption on all applications in cluster or rack or even the entire data centre level). The VPM states may or may not be actually supported by hardware but are a set of performance states for use by VMs application-specific management policies. The actual power management actions are carried out by the
350 infrastructure are defined as VPM rules and are realised by VMP mechanisms which include hardware scaling, soft scaling, and consolidation. Different from other related work to manage power at the level of physical hosts, *VirtualPower* enables fine-grained power control at the level of individual VMs. The biggest limitation of VirtualPower is that significant modifications to existing hyper-
355 visors must be performed, preventing it’s large scale deployment in existing infrastructures.

While VirtualPower only optimises energy efficiency on individual hosts, *pMapper* is a controller that places an application onto the most appropriate physical server in order to optimise energy and migration costs, while still meet-
360 ing some performance guarantees [41]. *pMapper* employs First Fit Decreasing (FFD) bin-packing algorithm to select an optimal server for any application being migrated in order to minimise power consumption. The algorithm optimises one major resource such as CPU utilisation and treats other resources such as memory and I/O as constraints. *pMapper* has three main modules: The
365 *Monitoring engine* monitors all VMs and physical servers and collects their performance (different workloads contribute to the overall CPU and memory utilisation) and power (overall usage) characteristics. The *Performance Manager* examines performance statistics collected from monitoring engine, produces a set of VM sizes that suit current loading and estimate potential benefits should
370 any VM resizing is required. The *Power Manager* keeps track of current power consumption and optimises it through CPU throttling. Nevertheless, *pMapper*

employs an *Arbitrator* to ensure consistency between the three modules. Subsequent works [56][33] focused on analysing real data centre application traces, and revealed that there are sufficient variations and correlations amongst applications to be exploited for improving power saving. Hence, *pMapper* has been
 375 extended to include some application-awareness features.

In contrast to *pMapper*, *Mistral* [42] is a system that emphasises on the optimisation of transient power/performance costs. In *Mistral*, application performance is reflected in application response time that is modelled as a layered
 380 queuing network (LQN). Power consumption of a configuration is based on an empirical non-linear model that concerns CPU utilisation (e.g., power consumption at idle and busy states). Different from *pMapper*, *Mistral* takes cost of six transient adaptation actions into account including: changes of a VM’s CPU capacity, addition and removal of a VM, live migration of a VM, shutting down
 385 and restarting a host, change in response time for the applications and change in power consumption during the adaptation. It use a workload predictor to predict the stability intervals of next adaptation based on historical average resource usage.

3.4. SLA-aware Consolidation Schemes

A Service Level Agreement (SLA) is a contract that sets the expectations, usually in measurable terms, between the consumer and service provider [57]. In a Cloud computing context, there are infrastructure SLA and service SLA. Infrastructure SLA is established between infrastructure (IaaS) providers and service providers to guarantee sufficient resource and uptime whilst service SLA
 395 is established between service providers and their customers and is typically measured in QoS metrics such as application response time: for example, maximum response time of 100 ms with minimal throughput of 100 transactions per second [57][58]. Since a SLA is the cornerstone of how the service provider sets and maintains commitments to the service consumer, optimising resource
 400 utilisation while not violating SLA is also crucially important for operators.

Bobroff et al. [36] propose a dynamic resource allocation algorithm for virtu-

alised server environments to maximise the global utilisation of the data centre,
 while not violating SLAs (i.e., VM’s CPU time guarantee) as a result of per-
 formance degradation due to overloading. Similar to other VM consolidation
 405 schemes, the algorithm collects and analyses historical usage data on resource
 utilisation and service quality and predicts the future demand, based on which
 a sequence migrations are computed. The algorithm (bin packing) is invoked
 periodically and thus forms a measure-forecast-remap (MFR) optimisation loop.
 For placement, the algorithm derives a minimum set of servers required to ac-
 410 commodate all VMs while not overloading the servers with SLA constraints, i.e.
 CPU time guarantee. This work, however, shares the similar limitation faced by
 prediction-based consolidation algorithms in which resource demand and future
 server resource have to be deterministic.

Breitgand et al. [43] present a Elastic Service Placement Problem (ESPP).
 415 Since SLA is defined as meeting the requirements of VM sizing, ESPP aims to
 optimally allocate variable sized VMs to physical hosts. In ESPP, each service
 is modelled as an application that spans over a set of VMs. Hence, the ESPP’s
 goal is to maximise the overall profit, which is measured with resulting VM size
 and probability of VM violation. However, this forms a generalized assignment
 420 problem (GAP) which is hard to solve. The authors relate ESPP to multi-
 unit combinatorial auctions and hence provide an integer program formulation
 of ESPP that is amenable to column generation. While column generation is
 efficient for VM placement in small data centres, it does not scale to large mega
 data centre.

425 3.5. *Network-aware Consolidation Schemes*

Meng et al. [48] propose a pioneering work in network-aware initial VM
 placement. The authors first studied two sets of real traffic traces in operational
 data centres and observed three key traffic patterns that can be exploited for VM
 placement: (a) Uneven distribution of traffic volumes from VMs; (b) Stable per-
 430 VM traffic at large timescale; and (c) Weak correlation between traffic rate and
 latency. Hence, VMs can be placed in a way that traffic is localised and managed

pairwise. In order to achieve these objectives, they formulated a minimisation problem based on a fixed cost of required bandwidth for each VM, and the cost of communication between the VMs. CPU and memory resources are not used
 435 in the algorithm since it is assumed that each VM has the same size and that each server supports a fixed number of VMs. They also showed that the VM placement as an optimisation problem is NP-hard. To reduce the complexity of the proposed algorithm, the authors use a scenario with constant traffic of grouped VMs to simplify the problem space. The VM placement problem is
 440 then solved using *min-cut* and *clustering* algorithms to find divisions for VMs.

Wang et al. [50], as an extension with dynamic traffic to [48], propose a VM consolidation scheme to match known bandwidth demands to server’s capacity limit. However, in contrast to classical Bin Packing optimisation in which network bandwidth demand is assumed to be static, the authors formulated a
 445 NP-hard stochastic bin-packing problem which models the bandwidth demands of VMs as probabilistic distributions and then solve it using an online packing heuristic that assumes each bin has unit capacity. They also assume that network bandwidth is only limited by host network devices rather than topological oversubscription. As only host bandwidth limit is considered, network capacity
 450 violations into the DC are possible because aggregation and core layer links are often oversubscribed.

Ballani et al. [5] tackled the unpredictability of network performance (network-awareness) with novel virtual network abstractions through which tenant virtual networks are mapped to operator’s physical networks using an online algo-
 455 rithm. The virtual network abstractions include a virtual cluster representing a topology that is comprised of a number of VMs, a non-oversubscribed virtual switch and a virtual oversubscribed cluster that reflects today’s typically oversubscribed two-tier cluster. Once the mapping is done, it is enforced through work (VMs) placement with a fast allocation heuristic - given a set of VMs (with
 460 bandwidth requirement) that can be placed in any sub-tree, the algorithm finds the smallest sub-tree that can fit all tenant VMs.

Unlike [48][50][5] which only consider network bandwidth constraint, Shri-

vastava et al. [59] proposed a framework which jointly considers inter-VM dependencies and underlying network topology for VM migration decisions. The
465 objective of the optimisation framework is to minimise the overhead (latency, delay, or number of hops) of migration by placing dependent VMs in close proximity in topological location. However, the problem is a variant of multiple knapsack problem and is thus NP-complete. An approximate algorithm, AppAware, is thus proposed in the paper.

470 Biran et al. [47] described a minimisation problem to determine the location of VMs in a data centre based on the network bandwidth, CPU, and memory resources. Their formulation is complex and does not scale to the size of data centres, thus they also created two heuristics based on the minimisation algorithm. The calculation is made off-line, and at every change it needs to
475 be executed. They assume that each user has a specific number of VMs that can only talk to each other, therefore all the VMs are already clustered in their approach.

As computation continues to shift from on-premises IT infrastructure into the Cloud, the computing platform now resides in a warehouse hosting (hundreds of) thousands of physical hosts. Today's Cloud DCs are no longer places
480 that house a large number of co-located servers, rather, they can be seen as a massive warehouse-scale computer [30]. The underpinning DC infrastructures are still a large-scale distributed system and therefore, one should consider converged, DC-wide resource optimisation rather than per-node-based optimisation. In particular, as data are constantly shuffled across the network, performance is ultimately limited by the network's aggregate capacity as bandwidth
485 is an expensive resource and is highly oversubscribed. It is therefore crucial that any resource management and optimisation scheme takes the network performance into consideration [60] [61][48][50][59][47][13].

490 Most of optimisation frameworks either rely on the prediction of future trends based on historical data, and proactively allocate resource based on predicted demand. Some employ directly measured metrics of interest, such as [60] [61], and dynamically adapt to changes according to measurement results.

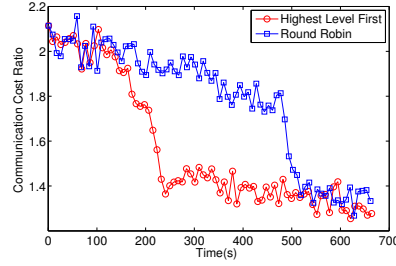


Figure 2: Granularity of spatial traffic demand measurement in S-CORE [60] has large impact on the performance.

Due to fluctuations in user demands [33][36][6], *direct measurement of tempo-*
 495 *ral resource usage seems more appropriate than prediction models in terms of*
exploiting resource availability in short timescales.

3.6. Open Research Issues

To demonstrate the flexibility of direct measurement, we have developed S-CORE [60][61][62], a distributed communication cost reduction VM migration
 500 approach which takes network cost into account. As opposed to aforementioned
 works in network aware server management, S-CORE employs a distributed
 algorithm based on information available locally through *direct measurement* of
 bytes exchanged at flow level at each VM to perform migration decisions, rather
 than using in-network or global statistics. This property allows the algorithm
 505 to scale and be realistically implementable over large-scale DC infrastructures.
 It iteratively localises pairwise VM traffic to lower-layer links where bandwidth
 is not as over-subscribed as it is in the core, and where interconnection switches
 are cheaper to upgrade.

Experimental results show that, by directly measuring traffic demand be-
 510 tween VMs, S-CORE can achieve significant (up to 87%) communication cost
 reduction, as shown in Fig. 2. The figure also highlights that S-CORE, when
 orchestrated with topology awareness (VMs whose traffic load is routed through
 the highest-layer links of the network topology are prioritised over close-minded
 VMs), converges significantly faster than when a topology-agnostic round-robin

515 orchestration scheme is used. This demonstrates that the spatial granularity,
i.e., the flow level direct traffic measurement, provides useful instantaneous net-
work knowledge that helps improve decision making (reflected in the convergence
time).

4. Network Resource Management

520 For the majority of applications hosted over Cloud environments (e.g., web-
indexing, distributed data analysis, video processing, scientific computing), data
is continuously transmitted over the network to support distributed processing
and storage as well as server-to-server communication [63]. These data-intensive
or latency-sensitive applications are particularly vulnerable to volatile through-
525 put and packet. Yet, the increased oversubscription ratios from bottom to the
top of prominent multi-root tree network architectures can result in poor server-
to-server connectivity hindering application performance [6][16].

Research has demonstrated that supporting protocols have failed to lever-
age topological advantages of new “scale-out” architectures [64][65]. Most no-
530 tably, recent measurement work [8][9][66] suggests that current DC networks are
largely under-utilised and therefore there is significant room (i.e., up to 20% of
network capacity [67]) for operators to improve performance before considering
expanding their network infrastructure or upgrading to new fabrics if provi-
sioning is reinforced with a finer-grained control loop. Resource fragmentation
535 can become a performance barrier in DC, resulting in low server utilisation and
therefore lower RoI [9][4][16][6].

Fine-grained network resource provisioning requires knowledge of the in-
stantaneous traffic demands, and subsequent harnessing of intelligent resource
admission control as well as exploiting the rich path redundancy of the under-
540 lying DC network. However, achieving such provisioning using existing legacy
mechanisms is faced with two fundamental challenges: First, estimating network
load based on historical traffic demands (i.e., predictions) is dubious, since these
change rapidly in DC environments and different patterns emerge over diverse

timescales [9]. Second, existing routing protocols such as ECMP fail to support
545 dynamic applications since they are load-agnostic and operate solely on packet
header contents [8][15].

4.1. DC Traffic Characteristics

Having a better understanding of traffic patterns can help in devising more
intelligent traffic management schemes that improve network performance. A
550 number of studies such as [8][9][68][69] have looked into Cloud DC traffic pat-
terns revealing some unique insights.

In a DC network, ToR Traffic Matrices (TM)s are sparse with significant lo-
cality characteristics, since a few ToRs exchange most data with just few other
ToRs [68]. Although a significant fraction of traffic appears to be localised
555 inside a rack, congestion does occur in various layers of the infrastructure de-
spite sufficient capacity being available elsewhere that could be used to alleviate
hotspots [8]. Congestion, when it happens, is shown to deteriorate application
performance by reducing server-to-server I/O throughput [9]. In terms of flow
distribution characteristics, data mining and web service DCs mostly accommo-
560 date small (mice) flows typically completed within 1s. Flow inter-arrival times
vary from 1 flow per 15 millisecond to 100 flows per millisecond at servers and
Top-of-Rack switches, respectively, while on average, there are 10 concurrent
flows per server active at any given time [9] [69] [6]. Finally, DC traffic patterns
change rapidly and maintain their unpredictability over multiple timescales (as
565 opposed to legacy Internet workloads), mainly due to the unpredictable dynam-
ics of external user requests as a result of resource sharing, and the multiplexing
of traffic at the level of individual flows, as opposed to large traffic aggregates [6].

Many Cloud applications follow Partition/Aggregate design patterns in which
application requests are divided into a number of smaller tasks which are then
570 distributed to a set of workers (servers). The intermediate results yielded from
these workers are aggregated to produce a final result. As a result, DCs mainly
run host applications with a multi-layer partition/aggregate pattern workflow
which exhibits pronounced Partition/Aggregate traffic patterns which exhibit

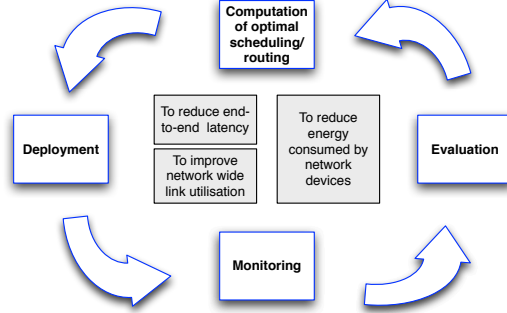


Figure 3: Traffic engineering is a procedure that optimises network resource utilisation through reshaping of network traffic.

bursty traffic patterns, resulting in Throughput Incast Collapse [70][71][72][73].

575 4.2. DC Traffic Engineering

Traffic engineering (TE) is a technique used by ISPs to select routes that make efficient use of network resources. More specifically TE is a procedure that optimises network resource utilisation through reshaping of network traffic. Fig. 3 illustrates a typical traffic engineering procedure and objectives that are
580 commonly used. TE consists of a control loop that continuously monitors and evaluates metrics of interest, based on which optimal resource scheduling is computed and deployed. TE techniques can be broadly classified as online and offline, the main distinction between the two being the timescales at which objective values, such as, e.g., link weights and scheduling of traffic flows are
585 adjusted.

Equal Cost Multipath (ECMP): In today’s data centres, Equal Cost Multipath (ECMP) is the most commonly used routing to spread traffic flows across redundant shortest paths using hashing on flow tuples (i.e., attributes of packet headers).

590 ECMP is easy to implement as it statically hashes one or more tuples of packet headers and subsequently schedules flows based on their hashed values, ensuring that packets of the same flow are all scheduled over the same path. A

commonly used 5-tuple hashing is based on *protocol identifier*, *source/destination network address*, and *source/destination port*.

595 **ECMP Challenges:** Recent research has shown that ECMP fails to efficiently leverage path redundancy in DC networks. Studies have demonstrated that network redundancy cannot completely mask all failures, implicitly pointing to the inefficiency of ECMP [66]. Similarly, it is shown that ECMP’s static hashing does not take either current network utilisation or flow size into consideration. Such hashing causes flow collisions that saturates switch buffers and
600 deteriorates overall switch utilisation, resulting in reduction in the network’s bandwidth [15]. Moreover, MicroTE [10] has tested ECMP with real DC traffic traces and found that ECMP achieves only 80-85% of optimal performance that can be obtained by solving a linear program with the objective of minimising the
605 Maximum Link Utilisation (MLU), assuming full prior knowledge of the traffic matrix every second. The implication of such inefficiency is that, while most of the links in measured DC networks have relatively low utilisation, a small but significant fraction of links appear to be persistently congested [8][9]. As a result, operators will need to upgrade their networks even if they are generally
610 under-utilised.

4.3. Utilisation-aware Traffic Engineering

Hedera [15] is a centralised TE mechanism aiming to resolve ECMP’s inability to fully utilise network bandwidth. Hedera’s is comprised of three steps. First, large flows (exceeding 10% of the host-NIC bandwidth) detection and
615 scheduling is carried out at the edge switches. Mice flows are still admitted using ECMP. Next, it estimates the natural demand, which is defined as the rate it would grow to in a fully non-blocking network, of large TCP flows. Based on the demand matrix, Hedera uses either global fit or simulated annealing heuristic placement algorithms to find best appropriate paths for different flows.
620 Eventually, these computed paths are pushed onto the switches. In contrast to only scheduling large flows, VL2 [6] uses Valiant Load Balancing to randomise packet forwarding on a per-flow basis. In VL2, two types of IPs are employed.

All switches and interfaces are assigned *location-specific* and applications use *application-specific* IP addresses, which remain unchanged regardless of server
625 locations as a result of VM migrations. Since each server randomly selects a path for each of the flow through the network, it shares intrinsic traffic-agnostic nature of ECMP.

MicroTE [10] is a fine-grained TE approach for DCs that achieves traffic adaptation by exploiting the short-term and partial predictability of the DC
630 traffic demands, to attain overall better link utilisation than ECMP. MicroTE has a centralised controller to gather network demands from the network and maintains a global view of network conditions. A bin-packing heuristic is then employed to find minimum cost path for a given set of (stable) traffic demands. However, unpredictable nature of traffic pattern in production DCs [6] puts
635 MicroTE’s usability under question.

The Modified Penalizing Exponential Flow-spliTing (MPEFT) [67] implemented and evaluated an online version of PEFT [74] to provide close to optimal TE for a variety of DC topologies by both shortest and non-shortest paths with exponential penalisation. MPEFT implements a hardware component in
640 a switch to actively gather traffic statistics and link utilisation which are then aggregated to a traffic optimiser. Similar to MicroTE, MPEFT is an online TE that optimises network resource utilisation in short timescales. Different from other schemes, MPEFT does not rely on static predictions of traffic demands which is proven to be unreliable [9]. Rather, MPEFT monitors traffic demands
645 in order to capture temporal traffic variability and then recomputes and schedules traffic to adapt to such variance. However, near optimality is achieved only when per-packet based scheduling is used.

4.4. *Energy-aware Traffic Engineering*

Cloud DCs are amongst major consumers of electricity and the trend is set
650 for it to rise even higher. It is estimated that amongst each Watt consumed, IT equipment takes about 59% of the share, 33% is attributed to cooling, and 8% is due to power distribution loss [4]. In order to reduce the energy con-

sumed by network equipment, energy-aware routing has been proposed using path diversity to conserve energy. For example, some schemes use as few network devices as possible to provide the routing service without compromising network performance [75]. Once, the minimum required set of networking nodes has been established, remaining idle ones can be shut-down or put to sleep mode to save energy. However, if the fault-tolerance is not considered, this approach can decrease the resiliency of the network under failure.

ElasticTree [76] is such an optimiser. It continuously monitors the DC's traffic conditions and then determines a set of network elements that must be powered on to meet performance and fault tolerance requirements; Switches or individual ports/links that are not needed can be shutdown. ElasticTree consists of three logical modules: optimiser, routing, and power control. The optimiser takes the topology, traffic matrix and a power model as well as the fault tolerance properties (e.g. spare capacity) as inputs to find minimum set of network that meets current traffic conditions. The output of the optimiser is a set of active components to both the power control and the routing modules. The power control is responsible for toggling the power states of switches, ports and line cards. The routing is responsible for flow admission and installs the computed routes into the network.

4.5. Latency-aware Traffic Engineering

High-bandwidth Ultra-Low Latency (HULL) [77] is an architecture that is designed for delivering predictable ultra-low latency and high bandwidth utilisation in a DC environment. In order to achieve this goal, HULL uses a combination of three technique: It uses Phantom Queues, which simulate the occupancy of a queue that drains at less than the maximum link rate, adaptive response to ECN marks using DCTCP [71], and packet pacing to smooth out bursts of packet arrivals. From both testbed and simulation experiments, it is reported that HULL mitigates tail latency by a factor of up to 10-40% through trading off network work throughput [77]. In other words, HULL does not eliminate queuing delay, but prevents it from building up.

Preemptive Distributed Quick (PDQ) [78] flow scheduling is a network protocol designed to improve flow completion time in order to meet deadlines. PDQ
685 borrows some key ideas from legacy real-time scheduling: use Earliest Deadline First to schedule tasks if they need to meet deadlines or use Shortest Job First if flow completion time is of higher priority. PDQ consists of a *PDQ sender* and a *PDQ receiver*. A *PDQ sender* sends a SYN packet to initialise a new flow and a TERM packet to terminate a flow; it is also for retransmitting a packet
690 if a timeout occurs. Whereas a *PDQ receiver* extracts the PDQ scheduling header from each data packet to ACK packets. However, since *PDQ* scheduling is a protocol that is fundamentally different from standard protocols existing in production switches, it can only work with custom-made PDQ switches. The PDQ switches share a common flow comparator, which assesses flow criticality
695 in order to approximate a range of scheduling disciplines [78]. PDQ requires switches to perform explicit rate control and flow state maintenance, and hence is complex to implement.

DeTail [69] is a cross-layer scheme for cutting the tail of flow completion faced by DC network traffic flows. At the link layer, DeTail employs flow control to manage port buffer occupancies and create a loss-less fabric. Each switch
700 in the network individually detects congestion by monitoring ingress queue occupancy which is represented with a *drain byte counters*. When it exceeds a predefined threshold, the switch informs the previous hop to pause its transmission by sending a *Pause* message with the specified priorities. Similarly, when
705 the *drain byte counters* falls below the predefined threshold, the switch resumes transmission by sending a *Unpause* message to the previous hop. DeTail employs congestion-based load balancing at the network layer by admitting flows on to least congested shortest paths. In comparison to PDQ, DeTail only cuts the tail of flow completion time rather cutting mean completion time.

Fastpass [79] is a logically centralised arbiter which allows end hosts to send
710 at line-rate while eliminating congestion at switches. This is achieved by taking packet forwarding decision out of end hosts and carefully schedule all flows in a time-sharing fashion, such that each host gets a small fraction of time to

use the network exclusively. The centralised arbiter also consists of a path
715 selection that scatters packets across all available links such that queues will
not build up. In comparison with PDQ, and DeTail, Fastpass does not require
hardware modification, but needs high precision clock synchronisation and will
increase the mean flow completion time due to the communication delay with
the controller for every packet in the flow.

720 Silo [80] provide cloud applications guaranteed network bandwidth, guar-
anteed packet delay and guaranteed burst allowance in order to ensure pre-
dictable network latency for their messages. Silo employs network calculus to
map such multi-dimensional network guarantees to queuing constraints on net-
work switches. Compared with other systems, Silo does not requires substantial
725 changes to hosts or network switches, and hence is readily deployable. However,
Silo still relies on the predictability of future demand and make static allocation
of bandwidth share.

4.6. Policy-aware Traffic Engineering

All networks, including data centre networks, are governed by network poli-
730 cies. Network policy management research to date has either focused on devising
new policy-based routing/switching mechanisms or leveraging Software-Defined
Networking (SDN) to manage network policies and guarantee their correctness.
Joseph et al. [81] proposed *PLayer*, a policy-aware switching layer for DCs con-
sisting of inter-connected policy-aware switches (*pswitches*). Middleboxes are
735 placed off the network path by plugging them into pswitches in PLayer. Based
on policies specified by administrators, pswitches can explicitly forward different
types of traffic through different sequences of middleboxes. PLayer is efficient
in enforcing network policies but it does not consider load balancing which is
widely used in today’s data centres.

740 Vyas et al. [82] proposed a middlebox architecture, CoMb, to actively consol-
idate middlebox features and improve middlebox utilization, reducing the num-
ber of required middleboxes for operational environments. Policy-Aware Appli-
cation Cloud Embedding (PACE) [83] is a framework to support application-

wide, in-network policies, and other realistic requirements such as bandwidth
745 and reliability. However, these proposals are not fully designed with VMs migration in consideration, and may put migrated VMs on the risk of policy violation and performance degradation.

Recent developments in SDN enable more flexible middlebox deployments over the network while still ensuring that specific subsets of traffic traverse the
750 desired set of middleboxes [84]. Kazemian et al. [85] presented *NetPlumber*, a real-time policy-checking tool with sub-millisecond average run-time per rule update, and evaluated it on three production networks including Google’s SDN, the Stanford backbone and Internet2. Zafar et al. [86] proposed *SIMPLE*, a SDN-based policy enforcement scheme to steer DC traffic in accordance to policy
755 requirements. Similarly, Fayazbakhsh et al. presented FlowTags [87] to leverage SDN’s global network visibility and guarantee correctness of policy enforcement. While these proposals consider policy enforcement as well as traffic dynamism, they require significant network status updates when VM migrations happen.

SYNC [88] and PLAN [89] study the impact of correct policy implementation
760 in the dynamic VM migration environment where change of end-point could imply violation of network policies. Both schemes overcome the difficulty by jointly considering network demand of VMs and policy chaining requirement which demands specific network paths. The problem was modelled as a NP-hard stable matching problem. Scalable and fast online heuristic algorithms
765 have been proposed to approximate optimal solution.

4.7. Open Research Issues

Most TE approaches and schemes discussed in this section share a common overall objective: to provide predictable and high-bandwidth network under highly variable traffic demands while also meeting other criteria such as, e.g.,
770 energy consumption minimisation. The common underlying control loop includes *monitoring*, *detecting*, and *adapting* promptly to problematic link load, providing a model that reacts to adverse conditions such as congestion.

The transient load imbalance induced by load-agnostic flow admission can

significantly affect other flows using a heavily-utilised link that is common to
775 both routes. Flows contending for the bandwidth of the shared link are more
likely to create congestion which in turn causes packet drops for flows sharing
the same bottleneck link. In most TCP implementations, packet loss will trig-
ger packet retransmission when the retransmission timer expires or when fast-
retransmit conditions are met. This additional latency can be a primary contrib-
780 utor to degradation of network performance since the retransmission timeout is
a factor of 100 or more than the round trip time over a DC network environment.

*Traffic flows are usually shuffled over shortest paths between communicating
hosts. In some cases, however, selecting a non-shortest path can be advantageous
for avoiding congestion or routing around a faulty path/node [67].* The surveyed
785 proposals in this section only use multiple equal cost path in DC environment.
In comparison, *Baatdaat* [90] and MPEFT [67] opportunistically include non-
shortest paths for packet forwarding. However, finding flow routes in a general
network while not exceeding the capacity of any link is the *multi-commodity flow*
problem which is NP-complete for integer number of flows. Hence, the routing
790 algorithm might consider non-shortest paths constrained by *no more than n*
hops longer than the shortest path in practice because it does not significantly
increase computation complexity [90].

The performance of current DC networks can be significantly improved if
traffic flows can be adequately managed to avoid congestion on bottleneck links.
795 This can be achieved by employing more elegant TE to offload traffic from con-
gested links onto spare ones and alleviate the need for topological upgrades.
Measurement-based traffic engineering techniques such as *Baatdaat* [90] and
MPEFT [67] can play an essential role in response to the immediate load fluc-
tuations. In contrast to reactive traffic engineering such as MicroTE [10], *Baat-*
800 *daat* employs a *measure-avoid-adapt* proactive control scheme based on network
programmability. *Baatdaat* uses direct measurement of link utilisation through
dedicated switch-local hardware modules, and constructs a network-wide view
of temporal bandwidth utilisation in short timescales through centralised SDN
controllers. Subsequently, it schedules flows over both shortest and non-shortest

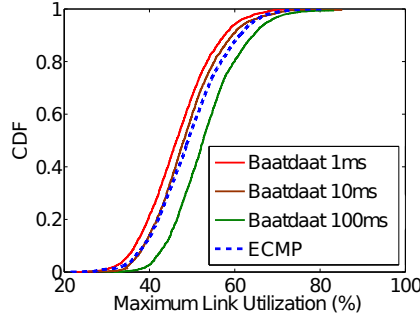


Figure 4: Granularity of temporal link load measurement (extracted from [90]) has large impact on the performance.

805 paths to further exploit path redundancy and spare network capacity in the DC. It is shown in [90] that direct measurement of link utilisation can help make better flow scheduling decisions which result in considerable improvement in maximum link utilisation. We reproduced in Fig. 4 the experimental results under different settings in *Baatdaat*. It can be seen that different measurement (and control) intervals can result in distinctively different performance results - *Baatdaat*'s performance gain over ECMP varies with the measurement timescale, and finer granularity yields better improvement. Even though the improvement is not uniform, in some regions can reach 20% over ECMP, while the practical measurement overhead is very low, especially if a dedicated management topology is used.

810

815

5. End-to-End Flow Control and Management

TCP is currently the most widely-used transport protocol carrying about 85% of the traffic on the Internet [91] and over Cloud DCs. Originally, TCP was designed for long-distance, Wide Area Network (WAN) communication with relatively long latencies and low bandwidth, however, DC characteristics are significantly different with Round Trip Times (RTT) below $250\mu s$, high throughput and a single administrative authority. Under these characteristics, TCP is known to under-utilise network bandwidth, leading in some cases to low through-

820

put and high latency [92][93]. To improve throughput, large buffers have been
825 used throughout the network reducing the number of retransmissions, however
large buffers cause side effects such as long latencies, traffic synchronisation as
well as preventing congestion avoidance algorithms to react promptly to congestion
events, leading to buffer-bloat [94]. TCP variants have been proposed to
enhance network utilisation over DC environments, however supporting existing
830 applications, workloads, and keeping the deployment complexity low proves to
be a challenging task.

5.1. Transport Protocols for Data Centre Networks

Most typical DC workloads such as search engines, data mining or distributed
file systems, follow the partition-aggregate paradigm where the work
835 is distributed amongst multiple machines and once each machine has computed
a partial result, this is aggregated back into a single point [8]. DC traffic generates
two types of flows: mice flows that represent 99% of the flows, are small
in size (less than a megabyte) and delay-sensitive; and elephant flows of aggregate
data carrying most of the bytes over the DC network. These large
840 flows are throughput-sensitive, bound by overall long completion times [71][8].
While mice flows are created by the query-response mechanism of the partition-
aggregate paradigm, elephant flows come from synchronisation mechanisms such
as distributed file system replication, database updates, and VM image migration.

845 One of the issues is that TCP’s conservative nature requires a constant value
for the Initial Window (IW) that cannot match the different environment requirements
from WAN to DC. If IW is smaller than the congestion window during congestion
avoidance phase, new flows will under-utilise the link until enough RTTs have
elapsed and the bottleneck capacity has been reached, or
850 the flow will terminate before exiting the *slow-start* phase. Over a fast DC
network with low latencies, the IW can overshoot the bottleneck capacity, triggering
packet loss and unfairness to other flows. Partition-aggregate patterns
generate bursts of ON-OFF traffic that can cause packets to be dropped or

delayed [8]. The conservative TCP parameters will wait some time before a
855 packet is retransmitted, however, the time for retransmission can be too long
for a packet to meet its deadline.

TCP has been shown to have significant issues in DCs mostly because
of its congestion avoidance mechanism. The low throughput under bursts of
flows and many-to-one communication is referred to as Throughput Incast Col-
860 lapse [70][92][72]. Due to this issue, Facebook reportedly switched to UDP in
order to have tighter control over the congestion mechanisms and avoid the ad-
verse impact of TCP on achievable throughput [71]. Facebook implemented a
UDP sliding window mechanism, with a window size inverse proportional to the
number of concurrent flows to solve incast collapse when receiving memcache
865 responses, halving the request time [95]. In order to fully utilise the DC network
infrastructure, new protocols have been designed to provide better performance,
maximise the throughput, minimise latency or reduce queue build-up.

A TCP enhancement, *GIP* [72], has recently been proposed to remedy TCP
incast throughput collapse. It has been identified that two types of time-outs
870 (TOs) termed Full window Loss TimeOut (FLoss-TO) and Lack of ACKs (Lack-
TO) are the major TOs that cause TCP incast problem. To avoid these TOs,
GIP reduces the congestion window at the start of each traffic burst (stripe
unit) and retransmits the last packet of every stripe unit for up to three times.
However, *GIP* does not deal with TOs due to packet loss and in high speed
875 environment with small buffers, the extra retransmission of the last packet can
needlessly increase buffer occupancy.

Data Center TCP (DCTCP) [71] leverages ECN in modern DCs to provide
a multi-bit feedback from a single-bit stream. Instead of treating each ECN-
marked packet as a congestion event, it uses the fraction of marked packets to
880 pace the sending rate. In doing so, the presence and extent of congestion can be
estimated. The main concept of this approach is to keep the buffer occupancy
of each switch as low as possible to prevent new packets from being delayed.
DCTCP requires support from the kernel in both the sending and receiving
hosts as well as Active Queue Management (AQM) with ECN support in the

switches. DCTCP has shown to be a significant step forward in preventing throughput collapse, however, it still reacts to an occurring congestion event instead of preventing such congestion to happen. DCTCP has now been included in Windows Server 2012 and Linux 3.18 as an alternative congestion control algorithm.

D^3 attempts to treat a DC as a soft real-time system, with each flow having deadline requirements with a revenue loss if it does not. It requires a new protocol that uses explicit rate control to apportion bandwidth according to flow deadlines. To calculate the rate of transmission, D^3 measures the number of flows traversing the interface using flow initiation and termination packets (SYN and FIN) [96]. D^3 is built on top of DCTCP. However, it has been shown in [97] that D^3 can make unfair bandwidth allocations.

TCP variants have been proposed to avoid queue build-up and therefore prevent high latencies. The Rate Control Protocol (RCP) [98] and the Variable-Structure congestion Control Protocol (VCP) [99] aim to estimate link congestion and avoid queue build-up, minimise flow completion time while being TCP-friendly. However, both these protocols require end-host and switch support. For long-distance, high-latency environments a significant number of protocols such as STCP, Fast TCP and XCP have been proposed. These protocols have opposite requirements to what is required in a DC. XCP uses a generalisation of ECN to have explicit feedback instead of using packet drops or the binary mechanism of ECN. Fast TCP estimate the base RTT of the network and uses this value as well as current RTT to estimate the current length of the buffers, the sending rate is adjusted with respect to the number of packets in the queues. These protocols have been optimised to achieve high throughput for long-lived flows over Long Fat Pipes (LFP).

One recent proposal to tackle latency is TIMELY [100], which reconsiders the applicability of (round trip time) RTT to estimate queue occupancy. RTT has not been considered in previous proposals because it is prone to noise such as system interrupts and processing in OS stack, given small – tens of microseconds – end to end delay in data centre environment. TIMELY overcomes this

limitation by using newly available hardware-assisted NIC timestamps to bypass OS stack. Once measured, TIMELY will compute the delay gradient, which reflects how quickly the queue is building or draining, and use it to compute target sending rate. TIMELY, on the other hand, requires fine-grained and high precision RTT estimation.

The issue with TCP is that it is complicated to keep high throughput with low buffer occupancy [71]. Without bloating in-network buffers, the end hosts must be able to pace the delivery of packets to match the characteristics of the link. However, such characteristics are commonly unknown by the end hosts and vary depending on the number of connected hosts and active concurrent flows. In order to establish that a packet has been dropped or lost, some algorithms such as F-RTO can be used but the last resort is to use timeouts. Such timers must be long enough not to worsen congestion by duplicating packets, but also short enough to avoid long delays between transmission and therefore low throughput.

The current trend in using commodity instead of DC-specific hardware shows that it is unlikely that application-specific hardware will be deployed in such infrastructures, hence algorithms requiring topology or hardware changes are unlikely to be deployed in production environments.

5.2. Open Research Issues

DC providers have full control over their infrastructure, allowing full network-wide knowledge of the topology, bandwidth, latency and network element properties (e.g., switch buffer sizes). Therefore, the default conservative values used to cope with the unknown characteristics of the Internet can be altered to match the network properties. Recently, with the wide deployment of Software Defined Networking (SDN) especially within DC networks [101], the current state of the network can be aggregated at a single or a hierarchy of controllers, and subsequently be used to distribute network knowledge to the end hosts in short timescales [28]. Amazon, Google and Microsoft showed a loss in revenue when response time increased by 100ms, creating a soft real-time constraint on the mice flows [102]. Because mice flows are delay-sensitive, it is necessary to pre-

vent the buffer occupancy of the switches to grow too large under traffic bursts as new flows will be delayed significantly. Due to inefficiency of TCP in DC networks, surveyed proposals concentrated on designing an alternative congestion control for TCP. In comparison, Omniscient TCP (OTCP) [103][73] tackles this
950 by exploiting SDN to tune TCP parameters for the operating environment.

With SDN, the TCP parameters can be tuned in real-time with respect to the current network state and prevent buffers from queuing up too much data. If the intra-DC Bandwidth Delay Product (BDP) is known alongside the number of flows on each link, the initial congestion window (IW) can be accurately
955 calculated to match the temporal network properties and increase network-wide throughput. Each flow IW has a fair slice of the network characteristics, and the total number of on-the-fly packets matches the BDP of the network with no buffer occupancy. Tuning IW based on the end-to-end intra-DC BDP, the amount of on-the-fly packets can be reduced to match the link properties and
960 hence reduce buffering. Such approach also prevents undershoot or overshoot of the IW size that can in turn lead to a long *slow-start* phase or packet loss in the first transmission, respectively.

Reducing in-network buffering and shortening *slow-start* will decrease the overall latency and improve the network utilisation. Buffering can be physi-
965 cally reduced by decreasing the size of SRAM in switches, also reducing hardware cost. However, with shallow buffers, throughput can be significantly lower under bursty traffic due to high number of packet drops and synchronised retransmissions (Incast Collapse). Carefully tuning the Minimum Retransmission Timeout (minRTO) allows high throughput to be achieved while keeping the
970 latency low [92][93].

Omniscient TCP (OTCP) [103][73] uses a SDN controller to keep the global state of the network and tune the minRTO and IW while a new route is being set up. This work shows that the bursty nature of DC traffic combined with large buffers and statically-assigned congestion control parameters, can signifi-
975 cantly delay and slow down the transfer of new incoming flows. However, solely reducing the buffer sizes can prevent high throughput from being achieved if the

default value of minRTO is used. Overall, the measurement-based IW estimation allows for reduced buffer occupancy and consequently bounds the latency; and a smaller minRTO allows throughput to be increased by pushing the congestion control logic back to the end-hosts.

6. Research Challenges and Opportunities

DCs are built on top of legacy hardware and software technologies currently deployed within ISP networks. Cloud operators often assume high similarity between the two environments and hence employ similar resource management principles - *static resource admission* and *over-provisioning* [15]. However, there are fundamental differences that are becoming apparent relatively early in the Cloud DCs' lifetime and will only intensify as their utilisation and commoditisation increase. The main ones relate to the level of aggregation at which resources are provisioned and managed, and at the provisioning timescales. Over the Internet, ISPs operate a relatively limited set of functions on traffic aggregates over long timescales. They can therefore rely on over-provisioning to accommodate short-term fluctuations in load, so long as aggregate demand is predictable over long timescales. On the contrary, Cloud DC operators manage a converged ICT environment where a plethora of diverse resources need to be provisioned over short timescales, and at a much finer granularity at the level of individual flows, links, virtual machine images, etc. The consequent demand is therefore highly unpredictable over both short and long timescales and DC operators need to respond to rapidly-changing usage patterns, as it has been demonstrated in a number of Cloud DC measurement studies [10][90].

At the same time, the collocation and central ownership of compute and network resources by a single Cloud service provider offers a unique opportunity for DC infrastructures to be provisioned in an adaptive, load-sensitive and converged manner, so that their usable capacity headroom and return on investment is increased, making Cloud computing infrastructures sustainable in the long term.

6.1. Network Control Plane Centralisation

Software Defined Networking (SDN) is a paradigm that allows a single control protocol to implement a range of functions such as routing, traffic engineering, access control and Virtual Machine (VM) migration [104][15][40][90].

1010 Network visibility is a unique feature of SDN, inherent to the central controller paradigm. Using such information, globally-informed decisions can be made that would not be possible in a legacy network where the control plane is fully distributed amongst the forwarding elements. A wide range of applications have been developed for SDN, such as, for example, network-wide middlebox
1015 and Access Control List (ACL) traversal. A middlebox is usually placed above the layer it must control, and adding a new one requires changes in the topology. Using SDN, middleboxes can be placed anywhere in the network, and traffic of the machines that must be controlled is redirected using SDN [84, 105, 106]. Such approaches may prevent shortest path routing, yet they allow the network
1020 to be much more flexible by programmatically enabling or disabling middleboxes and requiring no physical change in the topology when new features are added.

A large number of applications have been designed for SDN to demonstrate its benefits for network management and research. A number of studies already discussed in this paper have exploited SDN for resource management over DCs.
1025 For example, Hedera [15] uses OpenFlow to detect large flows in the network, estimate their demand and compute non-conflicting paths for them. In order to simplify VM migration, VL2 [6] and Portland [107] implemented novel layer 2 addressing and load balancing through OpenFlow.

6.2. Adaptive Data Centre Resource Management

1030 Traditionally, adaptive resource provisioning depends on the reliable prediction of future resource demand in order to estimate future requirements based on historical data sets. Such algorithms first collect a set of sample demands, and then compute a resource allocation (e.g., bandwidth or VM) optimisation. For example, adaptive and Dynamic Multi-path Computation Framework (ADM-
1035 PCF) [108] uses a large set of historical data to analyse and extract features from

traffic flows, which in turn are used to improve resource utilisation and mitigate congestion with guarantee on QoS. A prominent advantage of using this type of schemes is that they can potentially produce remarkable performance gain. It is shown that these algorithms can achieve near-optimal performance because
1040 when the resource requirement is relatively static, the collected set of sample when the resource usages deviate significantly from the anticipated normal behaviour (e.g., flash crowds, newly launched services, etc.), the resulting resource allocation can perform poorly [109][10]. Along with prediction-based provisioning, a more conservative yet costly provisioning approach is to over-provision
1045 resources to pre-empt peak demand [110].

6.2.1. Measurement-based Resource Management

There has been an implicit assumption that, similar to ISP networks, Cloud DCs exhibit stable traffic patterns over long timescales and that virtualisation provides performance isolation. However, such assumptions are increasingly challenged by measurement studies that demonstrate that performance
1050 interference does exist and that user demands change unpredictably causing most virtual machine management models to fail in achieving optimal allocation [34][35][111].

In order to overcome these challenges a more radical approach is required
1055 - *an approach that directly measures demand and resource utilisation, relaxing the need for unreliable prediction and costly over-provisioning.* Measurement of the activity on a target system reveals the utilisation. The utilisation characteristics of the system are then revealed without a specific, previous characterisation of the resource usage being known. Using measurements can remove
1060 the need for the complicated, and incomplete models of resource demand [112]. Measurement-based schemes can be used where no model is available. Often, when systems evolve and existing models are no longer suitable to capture their behaviour, measurements allow resource management to adapt to the evolving system.

1065 Measurements of resource utilisation need not be the only input to a man-

agement scheme. Measurement of other parameters such as I/O utilisation, ratio of VM admission, network latency, or the variation in server capacity among heterogeneous servers, are all examples of parameters which would provide important input to adaptive management schemes. After characterising resource usage, measurement-based management schemes can dynamically adapt to changes in resource requirements. Such dynamic and adaptive allocation of resources is simply not possible in environments where the imperfect models will result in static resource allocation or when prediction models fail. Measurement-based resource management will adapt resource allocation as usage patterns (and, consequently, resource demands) change. To enable the measurement of different parameters, physical servers, VMs and network nodes need to be appropriately instrumented to independently measure parameters of interest in a distributed manner.

A number of recent studies [60][113][103][67][90][49] have demonstrated that coupling distributed measurement with centralised decision-making is an approach that is able to offer significant improvement in resource provisioning adaptivity. Their results have shown that measurement-based techniques deviate only by a few percent from optimal, operate with less state and information, and offer new, adaptive services.

6.2.2. Open Issues

Measurement is a reactive method used to deal with rapid resource usage fluctuation. In order to capture the rapidly-varying user demand, the measurement intervals need to be small enough to characterise instantaneous change in user demand and resource utilisation [90]. The measurement process also needs to be fine-grained in order to measure resource usage at the same level as resources are leased, e.g., per-VM, per flow, per net-block [113] and then be able to shape admission as necessary.

However, fine-grained temporal and spatial control loops can be expensive because not only more capable and therefore expensive individual hardware is required but also the topology-wide control overhead increases [90][67].

Measurement-based resource shaping may be subject to control loop instability due to the sensitivity of adaptivity. High adaptivity can cause oscillations in resource allocation (e.g., in VM allocation or network routing algorithms) and penalise performance [114][115].

1100 Research efforts are needed to develop robust measurement mechanisms that take correct metrics into account and determine the fine balance between adaptivity and network-wide stability.

6.3. Converged Resource Management for Converged ICT

Virtualisation of server, storage and network resources has enabled the provisioning of converged infrastructures in which these components are virtualised and kept in a pool as a unified resource over DC environments. Resource provisioning for converged infrastructures is managed by a number of control loops at the routing [6][16][22][21][15], transport [116], and virtual machine (VM) [42][39][41] layers. Virtualised servers must be managed carefully to ensure the maximum utilisation of physical resources. Imperfect VM placement introduces unnecessary cross traffic that will either under-utilise or congest the aggregate and core network links by several factors since VM management algorithms only consider host-local NIC bandwidth constraints. An alternative is to also consider topological constraints such as aggregation links' capacity alongside conformance to server utilisation limits but this, currently, is still done using static information [49][50][113].

1105
1110
1115

Given the long retransmission timeout and other settings, it is different to manage transient congestion through all TCP variants. However, transient congestion is manageable if it can be detected and if alternate network paths can be exploited in short timescales [69]. For example, DCTCP [71] and HULL [77] can improve overall flow completion time to some extent but they are still unable to overcome the uneven load balancing as a result of static flow hashing due to lack of a global view of the link redundancy in the DC network, and hence still suffer from performance degradation. MPTCP is able to fairly split traffic across multiple TCP subflows but it lacks a mechanism to signal (and

1120
1125

guarantee) placement of subflows over physically diverse network paths.

Timely completion of Cloud application workloads heavily depends on the timely completion of network traffic flows. Network congestion is always the main cause of severe network performance degradation. The application layer
1130 has an abundance of information that can help the transport and network layers make better decisions. For instance, applications typically know the size of flows and whether they are latency-sensitive or latency-insensitive. By allowing applications to set some flow attributes such as priorities and sizes, more intelligence can be added to the network at different layers to ensure that every flow
1135 gets a fair share of the network bandwidth without compromising performance guarantee.

These examples demonstrate that the underlying DC infrastructure can significantly improve resource usage efficiency and performance through a coordinated, cross-layer approach that vertically spans all networking and systems
1140 layers. To this extent, we envisage a converged resource management framework for converged infrastructure environments: A unified control loop which will *measure*, *orchestrate* and *adapt* to enable the synergistic and dynamic allocation of network and server resources in order to achieve network-wide performance optimisation and offer predictable services even during short term, high
1145 utilisation fluctuations. Such synergy will include adaptive and load-sensitive server management, topology-aware traffic engineering, and network-informed traffic flow control. This will require the development of devices and mechanisms that will be programmable, will allow the exchange of information between the traditionally isolated layers of the software and network stack and, most impor-
1150 tantly, enable timely and measurement-informed service composition within the network itself.

6.3.1. The Role of DC Networks

DC networks will play a central role in the overall performance of Cloud computing environments since they provide the *central nervous system* for information exchange between cooperating tasks [117]. Moreover, the increased flat-
1155

tening of the Internet graph and the penetration of large-scale peering through public Internet exchanges, implies that end-users are only a few AS-hops away from any Cloud service provider and, therefore, the network-internal performance of these converged infrastructures becomes a significant contributor to the end-to-end service time [118].

This ubiquitous connectivity outstands the network as the ideal layer for the convergence of the control plane as it can provides a global view of activities from every node to every other node, whether these are servers or switches. Through the DC network, servers and network nodes can interact and disseminate node-local information such as resource provisioning and temporal consumption. By aggregating local knowledge, one can synthesise global performance views taking into consideration diverse metrics, to then apply innovative strategies to optimise the network-wide resource provisioning. Once measurement and control algorithms have computed network-wide objective functions, node-local (re)configuration options (and enforcement) can then be quickly disseminated over the DC network.

6.3.2. The Role of SDN

As measurement and control decisions are carried out by individual servers and network nodes in a distributed manner, SDN can evolve as *the brain* that centrally aggregates statistics and admits resources in a globally optimal (or approximate) manner. This logical centralisation of the control plane, facilitated through SDN, is necessary in order to avoid node-local optima resulting in globally-suboptimal resource admission (e.g., logical bandwidth oversubscription). At the same time, centralising the entirety of control algorithms can be detrimental: not only this would introduce new system vulnerabilities such as single points of failure, in some cases centralised computation of globally-optimal allocation problems is even computationally infeasible [60].

SDN can therefore play an important role in centralising network-wide orchestration and temporal resource information dissemination, yet distribute some of the control plane intelligence throughout the participating physical

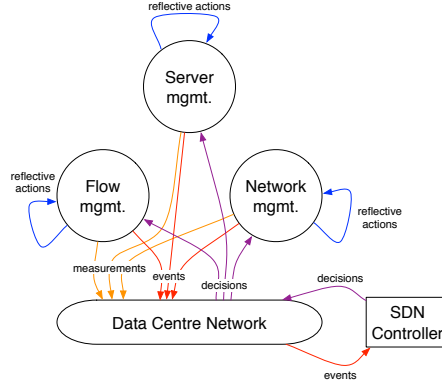


Figure 5: Converged resource management requires synergetic optimisation across application, network, transport and physical layers.

network entities that can synergistically analyse real-time workloads and subsequently adapt the network-wide operation gracefully to current load conditions.

Fig. 5 illustrates a framework for converged DC resource management. Each module individually measures node-local resource utilisation. Through the DC
 1190 network, these modules communicate events to a SDN controller and receive global decisions back. By integrating node-local intelligence with centralised control, management modules can effectively manage resource admission and shorten the decision-making time.

Nevertheless, realisations of SDN merely offer a match-commit framework
 1195 that can be programmable, yet all intelligence is centralised in a limited set of controllers that can easily bottleneck attempts to a converged resource management infrastructure. Recent development in SDN and dataplane programmability might alleviate this problem by allowing the network operators to dynamically reconfigure the dataplane. Using an abstract high level language such as
 1200 P4 [119] and a fast network oriented instruction set such as (e)BPF [120] to replace traditional match-action pipeline in switches, custom monitoring and telemetry modules could be available without performance impact on the network. In addition, current SDN protocol specifications do not offer any directives for host participation in a network-wide control plane. Further research is

1205 required to identify the fine balance between centralised control and distributed intelligence, cost, as well as in the areas of control theory and combinatorial utility function optimisation to support holistic and adaptive resource management for Cloud Data Centres.

7. Conclusion

1210 In this paper, we have provided a comprehensive and critical survey of resource management strategies for virtualised Data Centre (DC) infrastructures. We have structured this survey around the fundamental control loops that are typically employed to admit and manage resources, and leverage the infrastructural redundancy to boost performance while minimising operational costs. 1215 Diverse DC topologies have been presented that strive to exploit redundant connectivity to maximise the bisection bandwidth between any pair of servers using inexpensive equipment. We have surveyed strategies for Virtual Machine (VM) allocation and management to improve the utilisation and cost efficiency of physical servers, arguably the most costly investment for Cloud DC operators. We 1220 presented a body of research on resource-aware, energy-aware, network-aware and SLA-aware virtualisation management, and highlighted the fragmented and often conflicting objectives of the different schemes. We have moved onto surveying the management of DC network resources and presented documented evidence on the crucial role of the DC network on application performance. We 1225 have highlighted the challenges imposed by the distinct characteristics of DC traffic and emerging utilisation patterns, and presented developments on routing and flow scheduling mechanisms to improve utilisation, latency, and energy consumption over DC networks. Furthermore, we have discussed the main developments and implications of transport protocol design over DC networks, how 1230 congestion control can be adapted for high-speed, low-latency environments, and how multipath transport can be employed to leverage the underlying link redundancy.

Throughout this survey, we have highlighted the challenges imposed on man-

aging Cloud DC infrastructures due to the converged and colocated nature of
1235 these environments and the consequent co-existence of multiple disjoint control loops, mostly based on legacy mechanisms, each trying to optimise diverse and often contradictory objective functions. We then presented an extensive discussion on research opportunities for adaptive and converged resource management for virtualised DCs. We have highlighted the important role always-
1240 on, measurement-based provisioning can play in this, as well as the potential of Software-Defined Networking (SDN) as an orchestration framework for the converged shaping and allocation of virtualised resources over Data Centre infrastructures.

Acknowledgements

1245 The work has been supported in part by the UK Engineering and Physical Sciences Research Council (EPSRC) grants EP/N033957/1, EP/L026015/1, and EP/L005255/1.

References

- [1] Gartner Inc., Worldwide public cloud services market is forecast to reach
1250 \$204 billion in 2016 (2016).
URL <http://www.gartner.com/newsroom/id/3188817>
- [2] L. Columbus, Gartner predicts infrastructure services will accelerate cloud computing growth (2016).
URL <http://goo.gl/G58KH3>
- 1255 [3] R. Miller, <http://www.datacenterknowledge.com/archives/2009/05/14/whos-got-the-most-web-servers/> (2013).
- [4] A. Greenberg, J. Hamilton, D. A. Maltz, P. Patel, The cost of a cloud: research problems in data center networks, SIGCOMM Comput. Commun. Rev. 39 (1) (2008) 68–73 (Dec. 2008).

- 1260 [5] H. Ballani, P. Costa, T. Karagiannis, A. Rowstron, Towards predictable
datacenter networks, *ACM SIGCOMM Computer Communication Re-*
view 41 (4) (2011) 242–253 (2011).
- [6] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri,
D. A. Maltz, P. Patel, S. Sengupta, VL2: a scalable and flexible data
1265 center network, in: *Proc. ACM SIGCOMM’09*, 2009, pp. 51–62.
- [7] N. Farrington, A. Andreyev, Facebook’s data center network architecture,
in: *IEEE Optical Interconnects Conf*, IEEE, 2013.
- [8] T. Benson, A. Akella, D. A. Maltz, Network traffic characteristics of data
centers in the wild, in: *Proceedings of the 10th ACM SIGCOMM confer-*
1270 *ence on Internet measurement*, ACM, 2010, pp. 267–280.
- [9] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, R. Chaiken, The nature
of data center traffic: measurements & analysis, in: *Proc. ACM SIG-*
COMM Internet Measurement Conference (IMC’09), 2009, pp. 202–208.
- [10] T. Benson, A. Anand, A. Akella, M. Zhang, Microte: Fine grained traffic
1275 engineering for data centers, in: *Proceedings of the Seventh Conference*
on emerging Networking EXperiments and Technologies, ACM, 2011, p. 8.
- [11] BCN, Server capacity over-provisioned as cios bring legacy habits to cloud
platforms, 2014.
URL <http://goo.gl/cLftcj>
- 1280 [12] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Pat-
terson, A. Rabkin, I. Stoica, Above the clouds: A berkeley view of cloud
computing, Dept. Electrical Eng. and Comput. Sciences, University of
California, Berkeley, Rep. UCB/EECS 28 (13) (2009).
- [13] I. Diego, L.-P. Fabio, B. Bar’an, Many-objective virtual machine place-
1285 ment for dynamic environments, in: *2015 IEEE/ACM 8th International*
Conference on Utility and Cloud Computing (UCC), Limassol, Cyprus,
7-10 December 2015, IEEE, 2015.

- [14] Cisco Systems, Data center: Load balancing data center services solutions reference network design (2004).
- 1290 [15] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, A. Vahdat, Hedera: Dynamic flow scheduling for data center networks., in: The 7th USENIX Symposium on Networked Systems Design and Implementation (NSDI '10), Vol. 10, 2010, pp. 19–19.
- 1295 [16] M. Al-Fares, A. Loukissas, A. Vahdat, A scalable, commodity data center network architecture, ACM SIGCOMM Computer Communication Review 38 (4) (2008) 63–74 (2008).
- 1300 [17] R. Niranjan Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, A. Vahdat, Portland: a scalable fault-tolerant layer 2 data center network fabric, ACM SIGCOMM Computer Communication Review 39 (4) (2009) 39–50 (2009).
- [18] IBM RedBooks, IBM and Cisco: Together for a World Class Data Center, Vervante, 2013.
- [19] Cisco, Massively scalable data center at-a-glance (2010).
URL http://www.cisco.com/c/dam/en/us/td/docs/solutions/Enterprise/Data_Center/MSDC/1-0/MSDC_AAG_1.pdf
- 1305 [20] A. R. Curtis, S. Keshav, A. Lopez-Ortiz, Legup: Using heterogeneity to reduce the cost of data center network upgrades, in: Proceedings of the 6th International Conference, Co-NEXT '10, ACM, New York, NY, USA, 2010, pp. 14:1–14:12. doi:10.1145/1921168.1921187.
1310 URL <http://doi.acm.org/10.1145/1921168.1921187>
- [21] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, S. Lu, BCube: a high performance, server-centric network architecture for modular data centers, in: Proc. ACM SIGCOMM'09, 2009, pp. 63–74.

- 1315 [22] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, S. Lu, DCell: a scalable and fault-tolerant network structure for data centers, in: Proc. ACM SIGCOMM'08, 2008, pp. 75–86.
- [23] K. Chen, C. Hu, X. Zhang, K. Zheng, Y. Chen, A. V. Vasilakos, Survey on routing in data centers: insights and future directions, *Network*, IEEE 25 (4) (2011) 6–10 (2011).
- 1320 [24] L. Popa, S. Ratnasamy, G. Iannaccone, A. Krishnamurthy, I. Stoica, A cost comparison of datacenter network architectures, in: Proceedings of the 6th International Conference, Co-NEXT '10, ACM, New York, NY, USA, 2010, pp. 16:1–16:12. doi:10.1145/1921168.1921189. URL <http://doi.acm.org/10.1145/1921168.1921189>
- 1325 [25] P. Stalvig, Management networks – living outside of production. management networks segregate non-production traffic off production networks., Tech. rep., F5 Networks, Inc. (2008).
- [26] N. Feamster, J. Rexford, E. Zegura, The road to sdn, *Queue* 11 (12) (2013) 20 (2013).
- 1330 [27] Systems, Inc., Cisco safe reference guide, San Jose, USA, 2010.
- [28] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, Openflow: enabling innovation in campus networks, *SIGCOMM Comput. Commun. Rev.* 38 (2) (2008) 69–74, doi:10.1145/1355734.1355746 (Mar. 2008). doi:10.1145/1355734.1355746.
- 1335 URL <http://doi.acm.org/10.1145/1355734.1355746>
- [29] E. Saitović, I. Ivanović, Network monitoring and management recommendations (2011).
- 1340 [30] U. Hoelzle, L. A. Barroso, The datacenter as a computer: An introduction to the design of warehouse-scale machines, Morgan and Claypool Publishers (2009).

- [31] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, A view of cloud computing, *Commun. ACM* 53 (4) (2010) 50–58, doi:10.1145/1721654.1721672 (Apr. 2010). doi:10.1145/1721654.1721672.
URL <http://doi.acm.org/10.1145/1721654.1721672>
- [32] S. De Chaves, R. Uriarte, C. Westphall, Toward an architecture for monitoring private clouds, *Communications Magazine, IEEE* 49 (12) (2011) 130–137, doi:10.1109/MCOM.2011.6094017 (December 2011). doi:10.1109/MCOM.2011.6094017.
- [33] A. Verma, G. Dasgupta, T. K. Nayak, P. De, R. Kothari, Server workload analysis for power minimization using consolidation, in: *Proceedings of the 2009 Conference on USENIX Annual Technical Conference, USENIX'09*, USENIX Association, Berkeley, CA, USA, 2009, pp. 28–28.
URL <http://dl.acm.org/citation.cfm?id=1855807.1855835>
- [34] Y. Koh, R. Knauerhase, P. Brett, M. Bowman, Z. Wen, C. Pu, An analysis of performance interference effects in virtual environments, in: *Performance Analysis of Systems Software, 2007. ISPASS 2007. IEEE International Symposium on*, 2007, pp. 200–209. doi:10.1109/ISPASS.2007.363750.
- [35] P. Apparao, R. Iyer, X. Zhang, D. Newell, T. Adelmeyer, Characterization & analysis of a server consolidation benchmark, in: *Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, ACM, 2008, pp. 21–30.
- [36] N. Bobroff, A. Kochut, K. Beaty, Dynamic placement of virtual machines for managing sla violations, in: *Integrated Network Management, 2007. IM '07. 10th IFIP/IEEE International Symposium on*, 2007, pp. 119–128. doi:10.1109/INM.2007.374776.
- [37] J. Xu, J. Fortes, Multi-objective virtual machine placement in virtualized

- 1370 data center environments, in: IEEE/ACM GreenCom'10, 2010, pp. 179–
188. doi:10.1109/GreenCom-CPSCoM.2010.137.
- [38] D. Jayasinghe, C. Pu, T. Eilam, M. Steinder, I. Whally, E. Snible, Im-
proving performance and availability of services hosted on IaaS clouds
with structural constraint-aware virtual machine placement, in: IEEE
1375 Int. Conf. on Services Computing (SCC'11), 2011, pp. 72–79. doi:
10.1109/SCC.2011.28.
- [39] T. Wood, P. Shenoy, A. Venkataramani, M. Yousif, Black-box and gray-
box strategies for virtual machine migration, in: USENIX NSDI'07, 2007.
- [40] V. Mann, A. Kumar, P. Dutta, S. Kalyanaraman, VMFlow: Leveraging
1380 VM mobility to reduce network power costs in data centers, in: Proc.
IFIP TC 6 Networking Conf., Vol. 6640 of LNCS, 2011, pp. 198–211.
- [41] A. Verma, P. Ahuja, A. Neogi, pMapper: power and migration
cost aware application placement in virtualized systems, in: Proc.
ACM/IFIP/USENIX Int. Conf. on Middleware, 2008, pp. 243–264.
- 1385 [42] G. Jung, M. Hiltunen, K. Joshi, R. Schlichting, C. Pu, Mistral: Dy-
namically managing power, performance, and adaptation cost in cloud
infrastructures, in: IEEE Int. Conf. on Distributed Computing Systems
(ICDCS'10), 2010, pp. 62–73. doi:10.1109/ICDCS.2010.88.
- [43] D. Breitgand, A. Epstein, SLA-aware placement of multi-virtual ma-
1390 chine elastic services in compute clouds, in: IFIP/IEEE Int. Symp.
on Integrated Network Management (IM'11), 2011, pp. 161–168. doi:
10.1109/INM.2011.5990687.
- [44] VMWare, [http://www.vmware.com/products/vsphere/
features-drs-dpm](http://www.vmware.com/products/vsphere/features-drs-dpm) (2014).
- 1395 [45] Microsoft, [http://technet.microsoft.com/en-us/library/
gg610561.aspx](http://technet.microsoft.com/en-us/library/gg610561.aspx) (2014).

- [46] CITRIX, <http://www.citrix.com/products/xenserver/overview.html> (2014).
- [47] O. Biran, A. Corradi, M. Fanelli, L. Foschini, A. Nus, D. Raz, E. Silvera, A stable network-aware VM placement for cloud systems, Proc. IEEE/ACM Int. Symp. on Cluster, Cloud and Grid Computing (CCGRID '12) (2012) 498–506 (2012).
- [48] X. Meng, V. Pappas, L. Zhang, Improving the scalability of data center networks with traffic-aware virtual machine placement, in: Proc. IEEE INFOCOM'10, 2010, pp. 1–9. doi:10.1109/INFCOM.2010.5461930.
- [49] V. Mann, A. Gupta, P. Dutta, A. Vishnoi, P. Bhattacharya, R. Poddar, A. Iyer, Remedy: Network-aware steady state VM management for data centers, in: Proc. IFIP TC 6 Networking Conf., Vol. 7289 of LNCS, 2012, pp. 190–204.
- [50] M. Wang, X. Meng, L. Zhang, Consolidating virtual machines with dynamic bandwidth demand in data centers, in: Proc. IEEE INFOCOM'11, 2011, pp. 71 –75. doi:10.1109/INFCOM.2011.5935254.
- [51] S. Mehta, A. Neogi, Recon: A tool to recommend dynamic server consolidation in multi-cluster data centers, in: Network Operations and Management Symposium, 2008. NOMS 2008. IEEE, IEEE, 2008, pp. 363–370.
- [52] M. Chen, H. Zhang, Y.-Y. Su, X. Wang, G. Jiang, K. Yoshihira, Effective vm sizing in virtualized data centers, in: Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on, 2011, pp. 594–601. doi:10.1109/INM.2011.5990564.
- [53] F. Hermenier, X. Lorca, J.-M. Menaud, G. Muller, J. Lawall, Entropy: A consolidation manager for clusters, in: Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, VEE '09, ACM, New York, NY, USA, 2009, pp. 41–50.

doi:10.1145/1508293.1508300.

1425 URL <http://doi.acm.org/10.1145/1508293.1508300>

- [54] D. M. Gutierrez-Estevez, M. Luo, Multi-resource schedulable unit for adaptive application-driven unified resource management in data centers, in: Telecommunication Networks and Applications Conference (ITNAC), 2015 International, IEEE, 2015, pp. 261–268.
- 1430 [55] R. Nathuji, K. Schwan, Virtualpower: coordinated power management in virtualized enterprise systems, in: ACM SIGOPS Operating Systems Review, Vol. 41, ACM, 2007, pp. 265–278.
- [56] A. Verma, P. Ahuja, A. Neogi, Power-aware dynamic placement of hpc applications, in: Proceedings of the 22Nd Annual International Conference on Supercomputing, ICS '08, ACM, New York, NY, USA, 2008, pp. 175–184. doi:10.1145/1375527.1375555.
- 1435 URL <http://doi.acm.org/10.1145/1375527.1375555>
- [57] E. Wustenhoff, S. BluePrints, Service level agreement in the data center, Sun Microsystems Professional Series (2002).
- 1440 [58] L. M. Vaquero, L. Rodero-Merino, J. Caceres, M. Lindner, A break in the clouds: towards a cloud definition, ACM SIGCOMM Computer Communication Review 39 (1) (2008) 50–55 (2008).
- [59] V. Shrivastava, P. Zerfos, K.-W. Lee, H. Jamjoom, Y.-H. Liu, S. Banerjee, Application-aware virtual machine migration in data centers, in: INFOCOM, 2011 Proceedings IEEE, 2011, pp. 66–70. doi:10.1109/INFCOM.2011.5935247.
- 1445 [60] F. P. Tso, K. Oikonomou, E. Kavvadia, D. P. Pezaros, Scalable traffic-aware virtual machine management for cloud data centers, in: Distributed Computing Systems (ICDCS), 2014 IEEE 34th International Conference on, IEEE, 2014, pp. 238–247.
- 1450

- [61] R. Cziva, D. Stapleton, F. P. Tso, D. P. Pezaros, Sdn-based virtual machine management for cloud data centers, in: Cloud Networking (Cloud-Net), 2014 IEEE 3rd International Conference on, IEEE, 2014, pp. 388–394.
- 1455 [62] R. Cziva, S. Jout, D. Stapleton, F. P. Tso, D. P. Pezaros, Sdn-based virtual machine management for cloud data centers, IEEE Transactions on Network and Service Management 13 (2) (2016) 212–225, doi:10.1109/TNSM.2016.2528220 (June 2016). doi:10.1109/TNSM.2016.2528220.
- 1460 [63] G. Wang, T. Ng, The impact of virtualization on network performance of Amazon EC2 data center, in: Proc. IEEE INFOCOM’10, 2010, pp. 1–9.
- [64] A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, S. Banerjee, Devoflow: scaling flow management for high-performance networks, in: ACM SIGCOMM Computer Communication Review, Vol. 41, ACM, 2011, pp. 254–265.
- 1465 [65] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, M. Handley, Improving datacenter performance and robustness with multipath tcp, ACM SIGCOMM Computer Communication Review 41 (4) (2011) 266–277 (2011).
- 1470 [66] P. Gill, N. Jain, N. Nagappan, Understanding network failures in data centers: measurement, analysis, and implications, in: ACM SIGCOMM Computer Communication Review, Vol. 41, ACM, 2011, pp. 350–361.
- [67] F. P. Tso, D. P. Pezaros, Improving data center network utilization using near-optimal traffic engineering, IEEE Transactions on Parallel and Distributed Systems 24 (6) (2013) 1139–1148 (2013).
- 1475 [68] S. Kandula, J. Padhye, P. Bahi, Flyways to de-congest data center networks, Proc. ACM HotNets, 2009.

- [69] D. Zats, T. Das, P. Mohan, D. Borthakur, R. Katz, Detail: Reducing the flow completion time tail in datacenter networks, *ACM SIGCOMM Computer Communication Review* 42 (4) (2012) 139–150 (2012).
- 1480 [70] A. Phanishayee, E. Krevat, V. Vasudevan, D. G. Andersen, G. R. Ganger, G. A. Gibson, S. Seshan, Measurement and analysis of tcp throughput collapse in cluster-based storage systems., in: *FAST*, Vol. 8, 2008, pp. 1–14.
- 1485 [71] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, M. Sridharan, Data center tcp (dctcp), *ACM SIGCOMM Computer Communication Review* 40 (4) (2010) 63–74 (2010).
- [72] J. Zhang, F. Ren, L. Tang, C. Lin, Taming tcp incast throughput collapse in data center networks, in: *Network Protocols (ICNP)*, 2013 21st IEEE International Conference on, 2013, pp. 1–10. doi:10.1109/ICNP.2013.6733609.
- 1490 [73] S. Jouet, C. Perkins, D. Pezaros, Otcp: Sdn-managed congestion control for data centre networks, in: *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2016.
URL <http://eprints.gla.ac.uk/114177/>
- 1495 [74] D. Xu, M. Chiang, J. Rexford, Link-state routing with hop-by-hop forwarding can achieve optimal traffic engineering, in: *Proc. IEEE INFOCOM'08*, 2008, pp. 466–474.
- [75] Y. Shang, D. Li, M. Xu, Energy-aware routing in data center network, in: *Proceedings of the first ACM SIGCOMM workshop on Green networking*, ACM, 2010, pp. 1–8.
- 1500 [76] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, N. McKeown, Elastictree: Saving energy in data center networks., in: *NSDI*, Vol. 3, 2010, pp. 19–21.

- 1505 [77] M. Alizadeh, A. Kabbani, T. Edsall, B. Prabhakar, A. Vahdat, M. Yasuda,
Less is more: Trading a little bandwidth for ultra-low latency in the data
center, in: Proc. of NSDI, 2012.
- [78] C.-Y. Hong, M. Caesar, P. Godfrey, Finishing flows quickly with pre-
emptive scheduling, ACM SIGCOMM Computer Communication Review
42 (4) (2012) 127–138 (2012).
- 1510 [79] J. Perry, A. Ousterhout, H. Balakrishnan, D. Shah, H. Fugal, Fastpass:
A centralized zero-queue datacenter network, in: Proceedings of the 2014
ACM conference on SIGCOMM, ACM, 2014, pp. 307–318.
- [80] K. Jang, J. Sherry, H. Ballani, T. Moncaster, Silo: Predictable message
latency in the cloud, in: Proceedings of the 2015 ACM Conference on
1515 Special Interest Group on Data Communication, ACM, 2015, pp. 435–
448.
- [81] D. A. Joseph, A. Tavakoli, I. Stoica, A policy-aware switching layer for
data centers, in: ACM SIGCOMM Computer Communication Review,
Vol. 38, ACM, 2008, pp. 51–62.
- 1520 [82] V. Sekar, N. Egi, S. Ratnasamy, M. K. Reiter, G. Shi, Design and imple-
mentation of a consolidated middlebox architecture., in: NSDI, 2012, pp.
323–336.
- [83] L. E. Li, V. Liaghat, H. Zhao, M. Hajiaghayi, D. Li, G. Wilfong, Y. R.
Yang, C. Guo, PACE: Policy-aware application cloud embedding, in: Pro-
1525 ceedings of 32nd IEEE INFOCOM, 2013.
- [84] A. Gember, P. Prabhu, Z. Ghadiyali, A. Akella, Toward software-defined
middlebox networking, in: Proceedings of the 11th ACM Workshop on
Hot Topics in Networks, ACM, 2012, pp. 7–12.
- 1530 [85] P. Kazemian, M. Chang, H. Zeng, G. Varghese, N. McKeown, S. Whyte,
Real time network policy checking using header space analysis, in:

USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2013.

- 1535 [86] Z. A. Qazi, C.-C. Tu, L. Chiang, R. Miao, V. Sekar, M. Yu, Simplifying middlebox policy enforcement using sdn, ACM SIGCOMM Computer Communication Review 43 (4) (2013) 27–38 (2013).
- [87] S. K. Fayazbakhsh, L. Chiang, V. Sekar, M. Yu, J. C. Mogul, Enforcing network-wide policies in the presence of dynamic middlebox actions using flowtags, in: Proc. USENIX NSDI, 2014.
- 1540 [88] L. Cui, R. Cziva, F. P. Tso, D. Pezaros, Synergistic policy and virtual machine consolidation in cloud data centers (2016).
URL <http://eprints.gla.ac.uk/112944/>
- [89] L. Cui, F. P. Tso, D. P. Pezaros, W. Jia, Plan: A policy-aware vm management scheme for cloud data centres (December 2015).
URL <http://eprints.gla.ac.uk/112857/>
- 1545 [90] F. Tso, G. Hamilton, R. Weber, C. Perkins, D. Pezaros, Longer is better: exploiting path diversity in data center networks, in: IEEE International Conference on Distributed Computing Systems (ICDCS), 2013, pp. 430–439.
- 1550 [91] D. Murray, T. Koziniec, The state of enterprise network traffic in 2012, in: Communications (APCC), 2012 18th Asia-Pacific Conference on, IEEE, 2012, pp. 179–184.
- [92] Y. Chen, R. Griffith, J. Liu, R. H. Katz, A. D. Joseph, Understanding tcp incast throughput collapse in datacenter networks, in: Proceedings of the 1st ACM workshop on Research on enterprise networking, WREN '09, ACM, New York, NY, USA, 2009, pp. 73–82. doi:10.1145/1592681.1592693.
1555
URL <http://doi.acm.org/10.1145/1592681.1592693>

- [93] V. Vasudevan, A. Phanishayee, H. Shah, E. Krevat, D. G. Andersen, G. R. Ganger, G. A. Gibson, B. Mueller, Safe and effective fine-grained tcp retransmissions for datacenter communication, in: Proceedings of ACM SIGCOMM, Barcelona, Spain, 2009.
- [94] J. Gettys, K. Nichols, Bufferbloat: Dark buffers in the internet, Queue 9 (11) (2011) 40:40–40:54, doi:10.1145/2063166.2071893 (Nov. 2011). doi:10.1145/2063166.2071893.
URL <http://doi.acm.org/10.1145/2063166.2071893>
- [95] R. Jeff, High performances at massive scale (2009).
URL <http://video-jsoe.ucsd.edu/asx/JeffRothschildFacebook.asx>
- [96] C. Wilson, H. Ballani, T. Karagiannis, A. Rowtron, Better never than late: Meeting deadlines in datacenter networks, in: ACM SIGCOMM Computer Communication Review, Vol. 41, ACM, 2011, pp. 50–61.
- [97] B. Vamanan, J. Hasan, T. Vijaykumar, Deadline-aware datacenter tcp (d2tcp), SIGCOMM Comput. Commun. Rev. 42 (4) (2012) 115–126, doi:10.1145/2377677.2377709 (Aug. 2012). doi:10.1145/2377677.2377709.
URL <http://doi.acm.org/10.1145/2377677.2377709>
- [98] N. Dukkupati, Rate control protocol (rcp): Congestion control to make flows complete quickly, Ph.D. thesis, Stanford, CA, USA, aAI3292347 (2008).
- [99] Y. Xia, L. Subramanian, I. Stoica, S. Kalyanaraman, One more bit is enough, IEEE/ACM Trans. Netw. 16 (6) (2008) 1281–1294, doi:10.1109/TNET.2007.912037 (Dec. 2008). doi:10.1109/TNET.2007.912037.
URL <http://dx.doi.org/10.1109/TNET.2007.912037>
- [100] R. Mittal, N. Dukkupati, E. Blem, H. Wassel, M. Ghobadi, A. Vahdat, Y. Wang, D. Wetherall, D. Zats, et al., Timely: Rtt-based congestion

control for the datacenter, in: Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, ACM, 2015, pp. 537–550.

- 1590
1595
1600
1605
1610
1615
1620
1625
1630
1635
1640
1645
1650
1655
1660
1665
1670
1675
1680
1685
1690
1695
1700
1705
1710
1715
1720
1725
1730
1735
1740
1745
1750
1755
1760
1765
1770
1775
1780
1785
1790
1795
1800
1805
1810
1815
1820
1825
1830
1835
1840
1845
1850
1855
1860
1865
1870
1875
1880
1885
1890
1895
1900
1905
1910
1915
1920
1925
1930
1935
1940
1945
1950
1955
1960
1965
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
2025
2030
2035
2040
2045
2050
2055
2060
2065
2070
2075
2080
2085
2090
2095
2100
2105
2110
2115
2120
2125
2130
2135
2140
2145
2150
2155
2160
2165
2170
2175
2180
2185
2190
2195
2200
2205
2210
2215
2220
2225
2230
2235
2240
2245
2250
2255
2260
2265
2270
2275
2280
2285
2290
2295
2300
2305
2310
2315
2320
2325
2330
2335
2340
2345
2350
2355
2360
2365
2370
2375
2380
2385
2390
2395
2400
2405
2410
2415
2420
2425
2430
2435
2440
2445
2450
2455
2460
2465
2470
2475
2480
2485
2490
2495
2500
2505
2510
2515
2520
2525
2530
2535
2540
2545
2550
2555
2560
2565
2570
2575
2580
2585
2590
2595
2600
2605
2610
2615
2620
2625
2630
2635
2640
2645
2650
2655
2660
2665
2670
2675
2680
2685
2690
2695
2700
2705
2710
2715
2720
2725
2730
2735
2740
2745
2750
2755
2760
2765
2770
2775
2780
2785
2790
2795
2800
2805
2810
2815
2820
2825
2830
2835
2840
2845
2850
2855
2860
2865
2870
2875
2880
2885
2890
2895
2900
2905
2910
2915
2920
2925
2930
2935
2940
2945
2950
2955
2960
2965
2970
2975
2980
2985
2990
2995
3000
3005
3010
3015
3020
3025
3030
3035
3040
3045
3050
3055
3060
3065
3070
3075
3080
3085
3090
3095
3100
3105
3110
3115
3120
3125
3130
3135
3140
3145
3150
3155
3160
3165
3170
3175
3180
3185
3190
3195
3200
3205
3210
3215
3220
3225
3230
3235
3240
3245
3250
3255
3260
3265
3270
3275
3280
3285
3290
3295
3300
3305
3310
3315
3320
3325
3330
3335
3340
3345
3350
3355
3360
3365
3370
3375
3380
3385
3390
3395
3400
3405
3410
3415
3420
3425
3430
3435
3440
3445
3450
3455

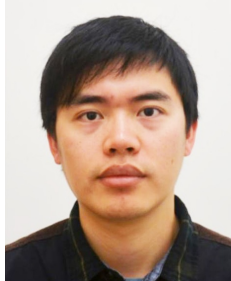
- [108] M. Luo, Y. Zeng, J. Li, W. Chou, An adaptive multi-path computation framework for centrally controlled networks, *Computer Networks* 83 (2015) 30–44 (2015).
- [109] H. Wang, H. Xie, L. Qiu, Y. R. Yang, Y. Zhang, A. Greenberg, Cope: traffic engineering in dynamic networks, *ACM SIGCOMM Computer Communication Review* 36 (4) (2006) 99–110 (2006).
- [110] R. Moreno-Vozmediano, R. S. Montero, I. M. Llorente, Elastic management of cluster-based services in the cloud, in: *Proceedings of the 1st workshop on Automated control for datacenters and clouds*, ACM, 2009, pp. 19–24.
- [111] W. Voorsluys, J. Broberg, S. Venugopal, R. Buyya, Cost of virtual machine live migration in clouds: A performance evaluation, in: *Cloud Computing*, Springer, 2009, pp. 254–265.
- [112] V. Lopez, H. Hamann, Measurement-based modeling for data centers, in: *Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*, 2010 12th IEEE Intersociety Conference on, 2010, pp. 1–8. doi:10.1109/ITHERM.2010.5501415.
- [113] F. Tso, G. Hamilton, K. Oikonomou, D. Pezaros, Implementing scalable, network-aware virtual machine migration for cloud data centers, in: *IEEE International Conference on Cloud Computing (IEEE CLOUD)*, 2013, pp. 557–564.
URL <http://eprints.gla.ac.uk/83096/>
- [114] R. Gao, C. Dovrolis, E. W. Zegura, Avoiding oscillations due to intelligent route control systems., in: *INFOCOM*, 2006.
- [115] S. Akhshabi, A. C. Begen, C. Dovrolis, An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http, in: *Proceedings of the Second Annual ACM Conference on Multimedia Systems, MMSys '11*, ACM, New York, NY, USA, 2011, pp. 157–168. doi:10.1145/1943552.

1943574.

URL <http://doi.acm.org/10.1145/1943552.1943574>

- [116] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, M. Handley,
1645 Improving datacenter performance and robustness with multipath TCP,
in: Proc. ACM SIGCOMM'11, 2011, pp. 266–277.
- [117] D. Abts, B. Felderman, A guided tour through data-center networking,
Queue 10 (5) (2012) 10 (2012).
- [118] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, W. Willinger,
1650 Anatomy of a large european ixp, in: Proceedings of the ACM SIGCOMM
2012 conference on Applications, technologies, architectures, and proto-
cols for computer communication, ACM, 2012, pp. 163–174.
- [119] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford,
C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, D. Walker, P4: Pro-
1655 gramming protocol-independent packet processors, SIGCOMM Comput.
Commun. Rev. 44 (3) (2014) 87–95, doi:10.1145/2656877.2656890 (Jul.
2014). doi:10.1145/2656877.2656890.
URL <http://doi.acm.org/10.1145/2656877.2656890>
- [120] S. Jouet, R. Cziva, P. Dimitrios P., Arbitrary packet matching in openflow,
1660 in: HPSR 2015, 2015.

1665



Fung Po Tso received his BEng, MPhil and PhD degrees from City University of Hong Kong in 2006, 2007 and 2011 respectively. He is currently lecturer in the Department of Computer Science at the Liverpool John Moores University (LJMU). Prior to joining LJMU, he worked as SICSA Next Generation Internet Fellow at the School of Computing Science, University of Glasgow. He

1670

has published more than 20 research articles in top venues and outlets. His research interests include: network measurement and optimisation, cloud data centre resource management, data centre networking, software defined networking (SDN), virtualisation, distributed systems as well as mobile computing and system.

1675

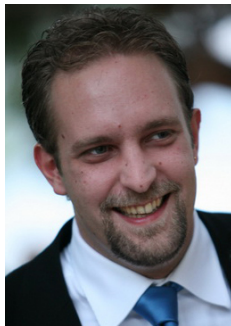


Simon Jouet received a M.Eng in Electronic and Software Engineering from the University of Glasgow in 2012. He is currently a Research Assistant at the School of Computing Science, University of Glasgow. His research focuses on the cross-layer benefits of centralized control in Cloud Data Centre in order to optimize resource utilisation, network and compute performance as well as energy

1680

efficiency. Current research focuses on the centralisation of network state, topology, routing and forwarding through Software Defined Networking (SDN) and orchestration through Network Function Virtualization (NFV).

1685



Dimitrios P. Pezaros received the B.Sc. (2000) and Ph.D. (2005) degrees in Computer Science from the University of Lancaster, UK. He is currently Senior Lecturer (Associate Professor) and director of the Networked Systems Research Laboratory (netlab) at the School of Computing Science, University of Glasgow, which he joined in 2009. His research is focusing on the resilient and effi-

1690

cient operation of future virtualised networked infrastructures through the exploitation of converged network-server resource management mechanisms, Software-Defined Networking (SDN), and Network Function Virtualisation (NFV). He has received significant funding for his research in the above areas from the UK Engineering and Physical Sciences Research Council (EPSRC), the University of Glasgow, the London Mathematical Society (LMS), and the industry.

1695

Previously, he has worked as a postdoctoral and senior research associate on a number of EPSRC and EU-funded projects in the areas of performance measurement and evaluation, network management, cross-layer optimisation, QoS analysis and modelling, and network resilience. Dimitrios has been a doctoral fellow of Agilent Technologies (2000-2004). He is a Chartered Engineer, and a Senior Member of the IEEE.

1700