Future Energy Efficient Data Centers with Disaggregated Servers

Article *in* Journal of Lightwave Technology · October 2017

CITATIONS
13

READS
500

4 authors:

Howraa M. Mohammad Ali
University of Kerbala
8 PUBLICATIONS **224** CITATIONS
SEE PROFILE

Taisir El-Gorashi
University of Leeds
289 PUBLICATIONS **4,606** CITATIONS
SEE PROFILE

Ahmed Q. Lawey
University of Leeds
67 PUBLICATIONS **2,156** CITATIONS
SEE PROFILE

Jaafar Elmirghani
University of Leeds
765 PUBLICATIONS **10,914** CITATIONS
SEE PROFILE

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

Future Energy Efficient Data Centers with Disaggregated Servers

Howraa M. Mohammad Ali, Taisir E. H. El-Gorashi, Ahmed Q. Lawey, and Jaafar M. H. Elmirghani

***Abstract*—The popularity of the Internet and the demand for 24/7 services uptime is driving system performance and reliability requirements to levels that today's data centers can no longer support. This article examines the traditional monolithic conventional server (CS) design and compares it to a new design paradigm: the disaggregated server (DS) data center design. The DS design arranges data centers resources in physical pools, such as processing, memory, and IO module pools, rather than packing each subset of such resources into a single server box. In this work, we study energy efficient resource provisioning and virtual machine (VM) allocation in DS-based data centers compared to CS-based data centers. First, we present our new design for the photonic DS-based data center architecture, supplemented with a complete description of the architectural components. Second, we develop a mixed integer linear programming (MILP) model to optimize VM allocation for the DS-based data center, including the data center communication fabric power consumption. Our results indicate that, in DS data centers, the optimum allocation of pooled resources and their communication power yields up to 42% average savings in total power consumption when compared with the CS approach. Due to the MILP high computational complexity, we developed an energy efficient resource provisioning heuristic for DS with communication fabric (EERP-DSCF), based on the MILP model insights, with comparable power efficiency to the MILP model. With EERP-DSCF, we can extend the number of served VMs where the MILP model scalability for a large number of VMs is challenging. Furthermore, we assess the energy efficiency of the DS design under stringent conditions by increasing the CPU to memory traffic and by including high non-communication power consumption to determine the conditions at which the DS and CS designs become comparable in power consumption. Finally, we present a complete analysis of the communication patterns in our new DS design and some recommendations for design and implementation challenges.**

***Index Terms*— Disaggregated Server, Data Center, Silicon Photonics, Energy Efficiency, Resource Provisioning, Data Center Communication Fabric.**

I. Introduction

VIRTUALIZED data centers provide key efficient services to clients with variable requirements. However, today's data center architectures are rigid in that they are composed of "server boxes", where each box has a predetermined ratio of CPU to memory to I/O that is unchangeable [1]. The single box server adds barriers and difficulties, including inefficient resource utilization, prolonged provisioning, and difficulties in big data management, as well as a high risk of blocking when deploying virtual data center resource instances. To understand the usefulness of the DS concept, consider the example of a conventional 'server in a box' (CS), where a processing intensive task occupies the processor while the input-output (IO) module is idle. Such an idle resource cannot be accessed in this case by other servers due to the current CS constrained architecture. Similarly, a server running an application involving intensive IO usage may have a large idle fraction of the CPU processing capability not accessible by other tasks that require access through the bottleneck IO module. The DS concept removes the barriers of the CS approach and allows VMs to construct servers on the fly with the required specifications for a specific duration and release these resources at the end of the task, thus removing many barriers and improving data center efficiency significantly.

Another challenge facing current data centers is the energy consumption of the physical infrastructure that provides resources for the cloud. Thus, energy management is a key challenge for data centers in reducing all their energy related costs [2, 3].

Significant efforts have been dedicated to optimizing the power consumption of conventional data centers, including energy efficient data center designs [4, 5], energy efficient inter- and intra-data center network architectures [6-8], designing energy efficient cloud computing services [9], [10], designing energy efficient network topologies [11], using renewable energy optimally [12], and introducing energy efficient resource provisioning and virtual network embedding for cloud systems [13].

The above work has a major shortcoming in that it relies on the single-boxed server approach, where flexible addition and removal of physical resources is very limited. Accordingly, in this paper, the disaggregated server (DS) architecture is considered as a potential approach to minimize data centers' power consumption. In this approach, servers' resources are separated into discrete pools of resources that are mixed and matched in real time to create differently sized and shaped systems. This technique brings a new server vision for data centers and motivates a plethora of potential new applications and services [14].

The revolutionary concept of DS can bring about radical change to traditional data centers and can simplify the vertical scalability of virtual machines (VMs) by decoupling the server

Manuscript received January 2017; revised ..... This work was supported by the Engineering and Physical Sciences Research Council (EPSRC), INTERNET (EP/H040536/1), and STAR (EP/K016873/1). The first author would like to support the Higher Committee for Education Development in Iraq (HCED Iraq) for funding her PhD. All data is provided in full in the results section of this paper.

The authors are from the School of Electronic and Electrical Engineering, University of Leeds, Leeds LS2 9JT, U.K.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

components from each other. On the other hand, resources are combined according to their types in a standalone and type homogenous "resource rack", constructing resource pools, where elements in the pool are interconnected using an optical backplane. Here, a data center network directly interconnects all resource racks via a high bandwidth and low latency inter-rack switching fabric [15]. Therefore, DS design introduces the sharing of CPU, memory, and network components' modularity and independent allocation of resources, such that a certain resource is no longer tightly coupled to any other resource, meaning that resources can be used more efficiently.

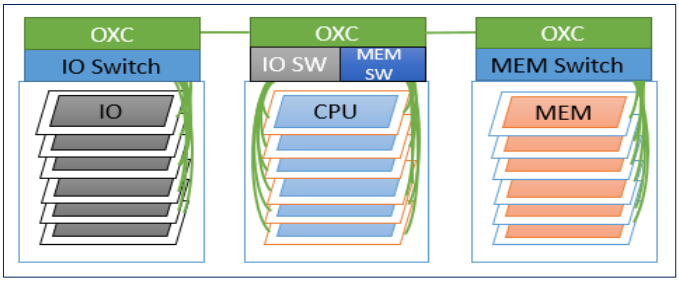


Fig. 1. DS concept.

Fig. 1 highlights the main concept of DS. In Fig. 1, we consider hybrid Electronic/Optical switching fabric in addition to the disaggregated resources pools. Therefore, resources in both the electronic IP layer (packet switched) and the optical layer (circuit switched) are needed. IP switches are needed to aggregate traffic from resources, and each IP switch is connected to an optical switch, which is connected to other optical switches by optical fiber links. Optical fibers provide the large capacity and fast data transmission required to support the communication between the disaggregated resources. Intel Silicon Photonic connectors (SiPh) [16] provide OEO processing for full wavelength conversion at each node. This architecture will be discussed in more detail in Section III, where we present the architecture of our DS design with a full description of all the components and communication patterns.

The main aim of this paper is to analyze the energy efficiency of the DS approach compared to CS using the MILP model and real-time heuristics. In addition, this paper gives a detailed description of the DS concept and provides a detailed architecture. In this work, we developed an energy efficient resource provisioning MILP model and simulation heuristic, considering the DS, and compared it to CS design. We accounted in our MILP model for the impact of the communication power on the total power saving and developed a heuristic which enables real time operation and verifies the MILP model. We have also designed a new switch-based communication architecture for the new DS-based data center.

It should be noted that the performance of the DS-based data center can be measured using several metrics and these include power consumption, latency, resilience, security, scalability and other metrics. It is not possible to study all these dimensions in a single article. Therefore our work reported here focuses on performance along the power consumption saving dimension. We however ensure that we serve all the VM requests, as serving a lower number of VM requests / rejecting VM requests results in unrealistic low power consumption and potential service level agreement violation. We also outline the improvements needed in electrical and optical switching technology to achieve the DS vision and outline further rack-clustering measures that can be introduced to help reduce latency. Performance along other known dimensions such as security, resilience and scalability, while interesting is outside the scope of this paper.

The remainder of this paper is organized as follows. Section II briefly reviews the related work. In Section III, we present our disaggregated server design. A description is presented in Section IV of the resource provisioning strategy in DS design with communication fabric. In this section, we introduce our MILP model for resource provisioning in DS, discuss its results, and propose the EERP-DSCF real-time heuristic. Section V presents a detailed evaluation of our disaggregated server design and the implementation of our proposed architectures. Finally, Section VI concludes the paper.

II. Related work

The idea of server disaggregation became prominent when pioneers, Intel and Facebook, announced their Open Compute Project [15-17] in the 2013 Open Compute Summit. Subsequently, a series of companies, including Cisco, Tencent, Mellanox, and some research groups, dedicated extensive time and effort to developing this topic. Before the 2013 Open Compute Summit, Mellanox Technologies [18] had shown that their Infiniband switching fabric can disaggregate the IO and storage subsystem, isolating these from the main computing system. The authors in [19-22] studied the DS idea and presented the design of a new general-purpose architectural building block, a memory blade, which allows memory to be "disaggregated" from the rest of the system ensemble. In [23] the authors discussed the ability of current data center communication networks to support the idea of disaggregation. The authors in [24] proposed a cloud architecture that disaggregates resources into virtual resource pools in order to provide virtual machines with the right amount of resources. Their cloud architecture creates a distributed and shared physical resource layer by providing a virtual layer and a cloud resource aggregation layer between applications and physical servers in real time. In [25], the authors presented Marlin, a memory-based addressing model for both I/O device sharing among multiple hosts and inter-host communications. Marlin is a PCI-based rack area network system which was designed to support the communications and resource sharing between disaggregated racks. In [26], Cisco presented the Cisco Composable Infrastructure, a software-defined infrastructure (SDI) solution which allows the infrastructure to be treated as a code, disaggregating compute resources so they can easily be programmed and automatically managed. A number of companies and university research groups have, in [27], provided their vision, work, and some metrics for the disaggregated data center. The authors in [28] and [29] presented an all-optical FPGA-based optical switch and interface card (SIC) for an optical programmable disaggregated data center network. In [30], a collaboration was set up between Tencent and Intel on a proof of concept project to demonstrate that the disaggregated data center and resource pooling, even in the early stages of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

development, can introduce improved performance and reduced power consumption and can enhance the end users experience. The authors in [31] studied the trade-off between cost and performance of building a disaggregated memory system; they constructed a simple cost model that compares the savings expected from a disaggregated memory system to the expected costs, such as latency and bandwidth costs, and then identified the level at which a disaggregated memory system becomes cost competitive with a traditional direct-attached memory system. In [32], the authors proposed a software-defined architecture for the next generation data center, dRedBox, and presented a design prototype hardware architecture where system-on-chip (SoC)-based microservers, memory modules, and accelerators are placed in separated modular server trays interconnected via a high-speed, low-latency optoelectronic system fabric and are allocated in arbitrary sets. In [33-35], we presented a detailed description and comprehensive review of the idea of DS, with a focus on the energy efficiency benefits of this design. We presented a summary of our MILP model and heuristics for resource provisioning and VM allocation in DS-based data centers and compared the performance of the new server approach to the CS-based data center. The idea of VM migration in DS was discussed and is augmented in this work with detailed results. Our contributions in this paper beyond the conference versions [33-35] are: (i) to the best of our knowledge, we are the first to develop a MILP model for energy efficient resource provisioning and VM allocation in disaggregated resources (CPU, memory, and IO) at the hardware level; (ii) we extend our MILP model in [33-35] to account for inter-rack communication power consumption; (iii) we develop simple and efficient heuristics that converge to near optimal solutions in a shorter time when compared with the MILP model; and (iv) we introduce a detailed architecture including a communications fabric that supports and enables interaction among the disaggregated resources.

III. Disaggregated Server Design

In this section, we present our photonically enabled design, which uses Intel's new photonic interconnect [16]. The design shares the memory and IO modules among multiple processors to form resource pools connected through a distributed switching fabric. The concept of distributed switch functionality and modular architecture design supports very granular resource deployment approaches, which allow for greater resilience and upgradability, and scaling up a VM can be done directly and seamlessly with this modular architecture. This architecture can potentially enable re-partitioning of the resources in such a way that system resources can be better shared between different compute elements.

Based on the ideas and guidelines given in [16-18] and [27], we built our modular architecture for the disaggregated server, and proposed a new interconnect topology to support the communication between the disaggregated server blocks. Given a data center system, the main communication components are inter- and intra-rack communications. Considering the inter-rack communications, the communicating units (e.g. servers or disaggregated devices) are located in different racks, while, for the intra-rack communication, these communicating units are located within the same rack. Thus, for DS, the communication that used to be confined inside single servers is now an inter-rack traffic and traverses the whole data center communication fabric.

To show the functionality of the suggested architecture and clarify its performance, we will define each type of these communications while describing our design. Moreover, we will show how each part of the architecture will perform its assumed function in supporting these communications. The following sections detail all the distributed components, focusing on each part of the architecture and the interconnecting components.

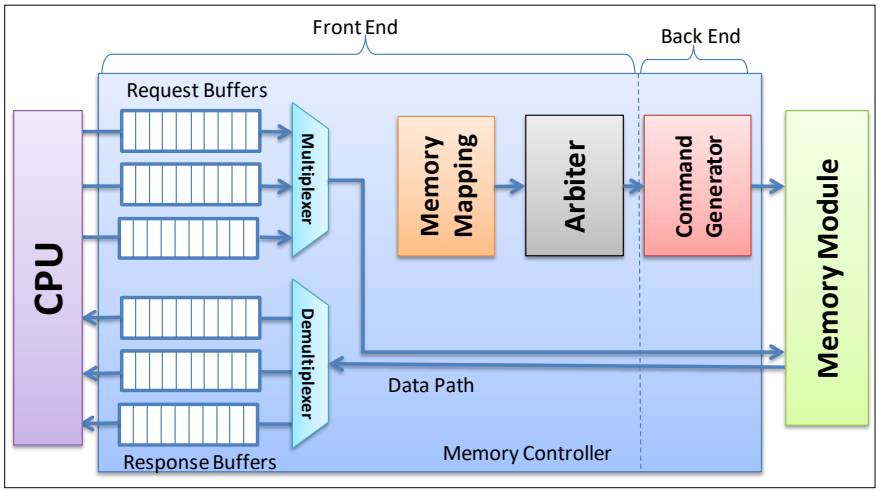
A. Disassembled Memory Controller (DMC)

The main driving factor we consider in the DS design is that resources are to be disaggregated while maintaining the same original interfaces they connected to before disaggregation. Based on this vision and design approach, the memory controller is disassembled into functional blocks to enable its disaggregation motivated by the fact that it has been moved over time from the motherboard's north bridge to other locations such as the CPU die. In this design, we split the memory controller into three functional blocks. The first block is attached to the CPU itself, named the CPU attached memory controller (CPUMC), and the second block is general to the whole memory rack, named the middling memory controller (MMC), while the last block is attached to the memory module directly and is the memory attached memory controller (MEMC). Before we present our new Disassembled Memory Controller (DMC), we need to examine the current classical memory controller. Fig. 2.a displays the complete architecture of the current memory controller [36]. It is mainly composed of two segments, the front end and the back end. While the front end is independent of the memory module type and provides an interface to the back-end segment of the memory controller, the back end is memory type dependent. It translates requests from the front end to the target memory.

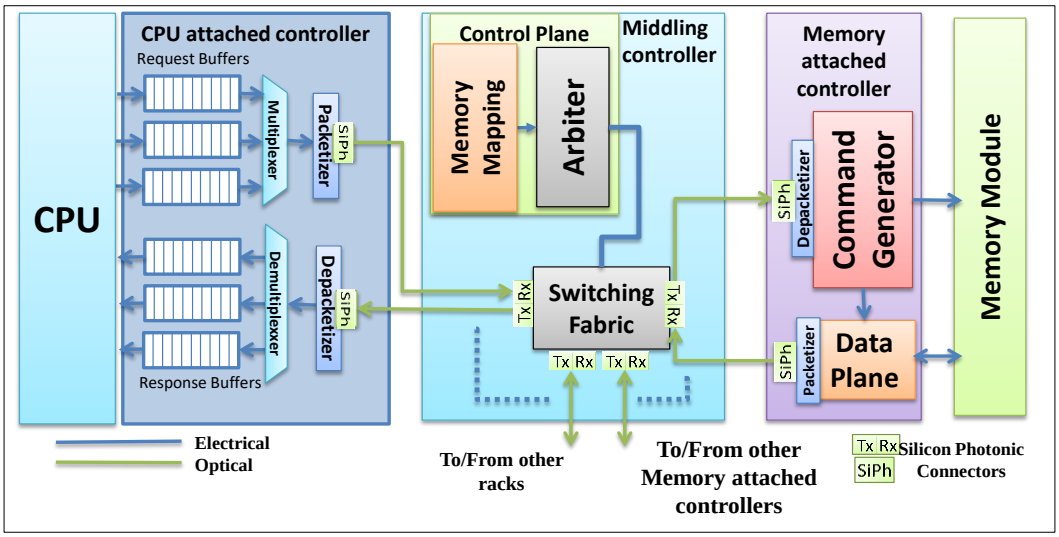
Functions such as buffering and instruction mapping and sequencing are performed in the front-end segment, which consists of buffers to store memory requests and responses. The buffers are attached to multiplexers/demultiplexers in order to send/receive one data word at a time [37]. The memory mapping decodes the memory address from the CPU address view to the memory address view (virtual memory to physical memory), and the arbiter decides the sequence in which requests from the CPU can access the memory modules. Thus, memory access requests are queued in the arbiter. The back-end command generator generates the commands for the target memory. It is memory-type dependent; thus, we will keep it attached to the memory, and it is customized to handle different timings so that different components having different clock rates can access the same memory module.

When disassembling the memory controller, we construct the three functional blocks shown in Fig. 2.b. The first block of Fig. 2.b, starting from the left, is the CPU directly attached to the CPUMC, as the CPU needs to see the same old interface. Buffers from the memory controller are attached to the CPU directly, and data is selected from these buffers in order to be sent to its destination memory rack.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology



(a) Classical Memory Controller



(b) Disassembled Memory Controller

Fig. 2. Memory controller.

In this block, we have added a packetizer [38], after multiplexing the incoming memory access requests from the CPU. The packetizer's role is to packetize the memory controller data to be switched between the CPU rack and the memory rack. On the other hand, the depacketizer puts the responses from the memory in normal data form to be read by the CPU. The block in the middle is the MMC, where the memory mapping and the arbiter functional blocks are integrated with the top of the memory rack switch. The memory mapping and the switch arbiter form the control plane of the switch. When receiving memory access requests, in packets form, the control plane of the memory controller reads the header of the packets and, according to the ID of the destination memory module, a path is established to the intended memory module. Finally, the command generator is attached directly to the memory modules to form the MEMC, as shown in Fig 2.b. It generates commands to read from/write to the memory through the control path for control signaling and through the data path for receive and send data.

B. Design Description

The architecture is wavelength division multiplexing (WDM) over hybrid optical and electrical switching, utilizing components such as the optical cross connect (OXC) switch and the electronic core packet (EXC) switch, in addition to Intel Silicon Photonic (SiPh) interconnect and optical fiber links, as shown in Fig. 3. In this architecture, Intel's new SiPh interconnect, which uses light as a speedy way to shuffle data between components, is used to perform the electrical to optical transformation function and to feed each fiber link. This new SiPh interconnect is designed especially for data center applications, using new materials and manufacturing techniques to be smaller, more resilient to dust and other pollutants, more reliable, and cheaper [38]. Our optically enabled modular architecture is a composition of different components, and the key ones are presented in Table I.

C. Racks Interconnect Topology

We have designed a modular software-defined architecture that can replace the traditional single rack of servers with three racks: the CPU rack, memory rack, and IO rack. These racks are connected and communicate using the new communication fabric described. In this architecture, our DS design is built up by disaggregating the server into its main components, where the switching between the racks is accomplished in a distributed manner through the use of the previously

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

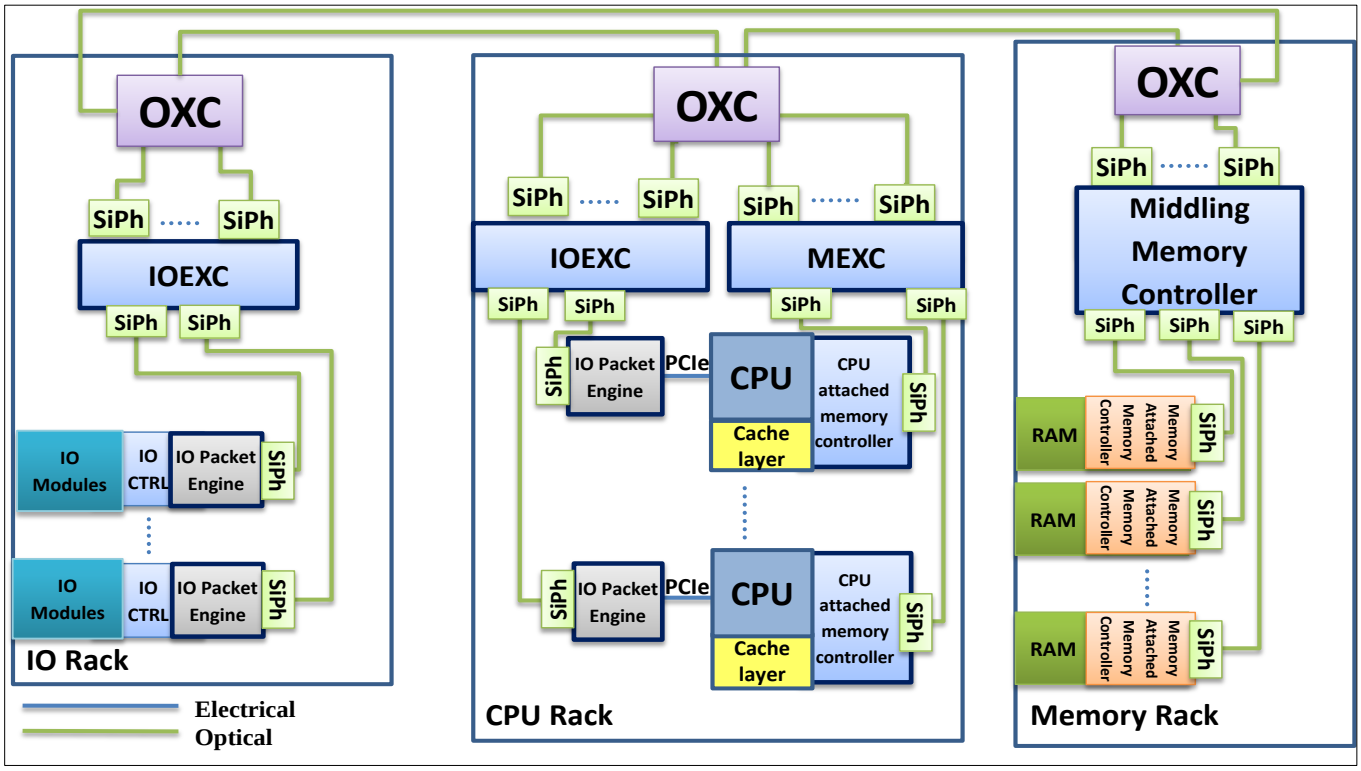


Fig. 3. DS architecture

mentioned components in Table I. Despite the fact that the single CS rack is replaced by three DS racks, power saving is still possible as the same number of resources (CPU, memory, IO, hard disks etc) are distributed (no increase in number of resources) and are utilized in an energy efficient manner in the DS racks as in [33]. In this work, DS racks consume extra power in the communication fabric, however, power saving is still possible as long as the increase in power in the communication fabric (DS communication fabric power minus CS communication power) is outweighed by the power saving due to efficient resource utilization as a result of the DS concept.

The DS design concept requires attention to the non-IT components that consume power. These non-IT components include networking, cooling and power supplies for example. The design of a custom networking architecture and subsystems for the DS has been considered in this work. The number of components to be cooled, i.e. CPUs, IO cards, and memory remain the same in the DS and CS cases (despite the fact that a single CS rack may be replaced by three DS racks). Therefore, the amount of cooling required in the data center remains the same provided custom cooling systems are introduced at the rack level instead of replicating the full server cooling system next to the CPU and replicating the full server cooling system next to (each of) memory and IO. The power supply unit has to be redesigned at the rack level instead of individual power supplies for each component (CPU, IO and memory). Such custom cooling and power supply designs for rack scale data centers are starting to appear in the industry [16]. Developments in this direction are already starting to appear in blade servers. Each server blade slides into a blade bay in a system chassis and plugs into a backplane to share common support components, such as power supplies, fans, CD-ROM, Ethernet and fibre channel and system ports [39].

Starting with the CPU rack, in this implementation, the new photonic interconnects and fiber cables are used to connect the CPUs throughout the rack via a point-to-point to a Top of Rack electronic memory switch (MEXC). These intra-rack connections are all optical, i.e. different wavelengths are used for the set of computing trays in each rack. In this design, the computing systems have been configured in trays, and each tray contains a single CPU die and its associated cache memory and control. Having large L1, L2, or even L3 caches reduces the impact of the main memory latency since the CPU can retrieve
cached data faster from its caches. The control consists of a CPUMC and PCIe interface connecting the CPU with the IO packet engine. Thus, both PCI and Ethernet networking protocols can be implemented in the same rack system, all enabled by the functionality of the MEXC and IOEXC switches, using light as the transmission medium over fiber channels.

TABLE I
MAIN COMPONENTS IN OUR DS DESIGN

Optical Connectors (SiPh)	Intel Silicon Photonic interconnects [16]
MEXC	Electronic switch that grooms CPU traffic to access RAM racks
IOXC	Electronic switch that grooms CPU traffic to access IO racks
IO Packet Engine	IO adapter [40]
IO CTRL	IO controller
OXC Switch	Top of Rack (ToR) optical switching units [22]
DMC Blocks	Disassembled memory controller blocks

Two IO packet engines are used in this design: one for each side of the CPU-IO link. This serial interface is configured to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

transfer the data and address and control information required to communicate with external IO modules, such as hard disks and Ethernet ports, using a serial packetized protocol. The CPU side IO packet engine provides an interface to the CPU and supports the IO switch IOEXC on top of the CPU rack by packetizing/depacketizing the IO control/data signals to be sent to their intended destination. The IO side IO packet engine provides an interface to peripheral devices, such as IO cards, to support the support the communication between the disaggregated resources. This relies on the design idea given in [18], where the IO modules are disaggregated from the rest of the server box.

Due to differences between the memory and IO packet formats, two separate switches have to be implemented: one for the CPU-MEM, and the other for the CPU-IO communications. Another reason for separating the electronic switches is that the CPU-MEM communication is latency intolerant. Here, application specific switches have to be used, which are normally expensive but are high performance, in contrast to the CPU-IO traffic, which is latency tolerant, and commodity switches can be used to transfer such traffic. Furthermore, the separation of the two types of traffic reduces the load on the bottleneck MEXC and results in fast communication. These switches are very important for traffic grooming in collecting traffic from different CPU cards to optimize the number of wavelengths used in the optical layer. These switches can be programmed to assign all traffic associated with a particular CPU to a specified port. The switch is programmable in order to allow software-based implementation of the protocols used for communications at any particular port. The output from these switches is fed into an OXC switch, which is the gateway for the rack to connect it with its neighboring racks. The connecting inter-rack links, linking the OXCs, are all optical in order to achieve a high bandwidth, low latency data transmission and simplicity of wiring through the use of fewer cables/fibers, which is an essential issue for certain dense applications.

The number of output ports of each WDM OXC switch depends on the number of neighbors of the rack where the switch resides, where these outputs are connected to their neighboring OXCs. In the memory rack, starting from the top, the OXC is connected to the middling memory controller via the fibers and silicon photonic interconnects. The middling memory controller, in turn, provides a path to the selected memory module. The middling memory controller combines both the switching and the MMC functionalities.

After switching, the data is sent to the MEMC, attached to the required memory module, optically. Additionally, our design supports direct memory access (DMA), such that the memory rack can communicate with the IO rack directly without interrupting the busy CPU. This is because the memory and IO racks are interconnected through their top of rack WDM OXC switches, either by passing the OXC on top of the intermediate CPU rack in the bypass scenario or through the intermediate top of CPU rack OXC, in a non-bypass scenario.

The IO rack structure is relatively similar to the memory rack, and it is disaggregated in a similar way to what is done in [18], with the use of the IOEXC to support the OXC. All the communication links here are optical to achieve fast and high bandwidth transmission. In this rack, the WDM OXC on the very top of the IO rack is connected to the electronic switch on top of the IO modules, IOEXC, and the IOEXC is connected optically to the different IO modules, which reside in the IO rack, through their packet engines and passing their IO controllers.

Communication integrity, control, and management are provided by a global data center operating system (GOS). This operating system is a general control layer that has an inclusive view for the whole disaggregated racks with their connectivity in order to be able to provide fluency in communication and manage the connectivity.

A hypervisor, which is a software layer that runs on top of the hardware resources and provides virtual partitioning capabilities to higher-level virtualization services, can be coupled with the GOS. The hypervisor enables the GOS to supervise and multiplex multiple operating systems in order to maintain and control every resource at all times and enable different operating systems to operate cooperatively.

CPU-to-CPU communication is managed by the top of memory rack electronic switch, MMC, and the MEXC, as communicating CPUs will interconnect through the remote memory modules, shared memory, they are using [41, 42]. CPU-MEM rack communication is performed by mutual functionality between the OXCs, DMC blocks, and the MEXC. The CPU-IO communication is facilitated by the functionality of IO packet engines on both racks to support the switching fabric implemented by the IOEXCs and the OXCs.

In brief, in the CPU rack, there are CPU trays whose traffic is aggregated using an electronic switch and is forwarded to the destined rack through optical layer switching using the OXC switch.

IV. Energy Efficient Resource Provisioning in DS Server with Communication Fabric

In the literature, a number of energy efficient inter-data center communication networks and architectures have been proposed and previously studied in [43-45]; however, data center energy management is still a hot topic for both industry and academia. We believe that implementing the DS-based data centers architecture can bring a variety of benefits, considering different prospects which include improved energy efficiency. In this section, we focus on the energy efficiency gains of resource provisioning and VM allocation in a DS-based data center. Data centers are large computing facilities which are built for applications that have very diverse resource requirements and are supposed to last for 15 to 20 years. Some applications are network intensive, such as video streaming applications, while others are latency sensitive and/or CPU intensive, such as web searching. The loads on a data center vary throughout the day and are related to our daily life events. This, in turn, creates challenges in attempting to reduce power consumption while maintaining the data center's performance. Precise resource provisioning and management directly influence overall data center energy efficiency and are of extreme importance in data center design. Under-provisioning of data center resources means that resources will be bottlenecked, while over-provisioning data center resources means a loss of power and capital. Thus,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

accurate provisioning is of vital importance and motivates the efficient design of data centers. Our vision is that implementing DS to provide a solution for the problem of good resource provisioning can result in notable outcomes.

Most of the previous work in the area of resource provisioning in data centers has focused on dealing with the VM itself, such as slicing [46], queuing, and migration [47], and multiple VM multiplexing [48]. In this paper, and due to the limitations of current server design, we study the DS approach as a means of improving data center resource provisioning and resource utilization. In this sense, the main aim is an efficient data center in terms of power consumption.

In the following paragraphs, we describe the type of data center we considered and how we can account for the power consumption associated with a requested VM running in the data center. We present details of the assumptions and system configuration for the resource provisioning and VM placement using disaggregated resources. Each VM request is identified to destination based on our novel design for the DS architecture. We present our MILP model, followed by a heuristic that mimics the MILP model behavior and expands the scope of the MILP model by providing lower complexity algorithms that enable real-time operation of the DS data center and enable the evaluation of relatively large size DS data center clusters.

A. *Resource Provisioning MILP model with Communication Fabric*

As explained in Fig. 1 and in Section III, each processing resource rack is served by two electrical switches, one for CPU-MEM communication and the other for CPU-IO communication; in addition, there is an optical switch on the top of the rack. The memory rack and IO rack are each served by a single electrical switch and a top of rack optical switch. All optical switches on top of the racks are connected in a semi- mesh connection. Inside each rack, the transceivers [49] shown in Fig. 3 (SiPh) support each port in each electronic switch. Each link is supported by transceivers and packetizers (packet engines for communications with IO modules) [50] at each end, one next to the source resource and one next to the destination resource. In addition, an optical Mux/Demux [51] is added after the transceivers at the link ends near the resources. As each transceiver is a 100 Gb/s in our design, and single resource traffic could exceed this 100 Gb/s, more transceivers can be used by a single resource, imposing the need to add multiplexing units. For the added functionalities of the memory mapping and arbiter to the MMC, we consider an additional 5 W to each working MMC to account for the power consumption of these units.

VMs demand resources in both the IP layer and the optical layer, in addition to the underlying DS resources. For evaluation, we define the following sets:

Sets:

NR	Set of all racks
PR	Set of CPU racks
MR	Set of memory racks
IOR	Set of IO racks
N_a	Set of neighbor racks of rack a
VM	Set of VMs to be served
NP	Set of CPUs in each CPU rack
NM	Set of memories in each memory rack
NIO	Set of IOs in each IO rack

by a unique id, denoted by index i; in addition, each CPU, memory, and IO module in the data center is similarly identified by a unique id, denoted by index j. Throughout the rest of the paper, we use VM and VM requests interchangeably to refer to requested resources by a VM.

In order to optimize the VM placement in DS-based data centers, consideration has to also be given to the inter-rack communication power consumption, which is very important due to the new DS design structure. In the CS data center, resource utilization may not be as efficient as in the DS data center; however, the traffic which used to be contained within the same server or the same rack in the CS data center, now typically navigates through several racks spanning part of the data center fabric.

In this section, we highlight the main components required to establish an end-to-end connection and guarantee a fast and durable communication path from source

The power consumption of a data center based on the DS architecture is composed of two parts, the first is the power consumed by active resources:

1) The power consumption of active processors

$$\sum_{p\in PR}\sum_{j\in NP}((XP_j^p \cdot Pmin) + (\Delta P \cdot \delta P_j^p))$$

2) The power consumption of active memories

$$\sum_{m\in MR}\sum_{j\in NM}((XM_j^m \cdot Mmin) + (\Delta M \cdot \delta M_j^m))$$

3) The power consumption of active IO modules

$$\sum_{io\in IOR}\sum_{j\in NIO}((XIO_j^{io} \cdot IOmin) + (\Delta IO \cdot \delta IO_j^{io})$$

The second part is the power consumed by networking elements:

1) Power consumption due to CPU-MEM traffic, which in turn is composed of:
   a) The power consumed by the electrical switches
   $$\sum_{p\,\in\,PR} PRS \cdot QPM_p + \sum_{a\in NR}\sum_{b\in N_a} PRI \cdot WPM_{a,b}$$
   b) The power consumed by the optical switch
   $$\sum_{a\in OPR} PO$$
2) Power consumption due to CPU-IO traffic, which is composed of:
   a) The power consumed by the electrical switch
   $$\sum_{p\in PR} PRS \cdot QPIO_p + \sum_{a\in NR}\sum_{b\in N_a} PRI \cdot WPIO_{a,b}$$
   b) The power consumed by the optical switch
   $$\sum_{a\in MR} PO$$
3) Power consumption due to Memory-IO traffic, which consists of:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

a) The power consumed by the electrical switch

$$\sum_{m\in MR} PRS\cdot QMIO_m + \sum_{a\in NR}\sum_{b\in N_a} PRI\cdot WMIO_{a,b}$$

b) The power consumed by the optical switch

$$\sum_{a\in IOR} PO$$

The MILP model is defined as follows:

Objective: minimize:

$$\sum_{p\in PR}\sum_{j\in NP}((XP_j^p\cdot Pmin)+(\Delta P\cdot \delta P_j^p))+$$
$$\sum_{m\in MR}\sum_{j\in NM}((XM_j^m\cdot Mmin)+\Delta M\cdot \delta M_j^m))+$$
$$\sum_{io\in IOR}\sum_{j\in NIO}((XIO_j^{io}\cdot IOmin)+\left(\Delta IO\cdot \delta IO_j^{io}\right)$$
$$+$$
$$\sum_{p\in PR} PRS\cdot QPM_p+$$
$$\sum_{a\in NR}\sum_{b\in Na} PRI\cdot WPM_{a,b}+$$
$$\sum_{p\in PR} PRS\cdot QPIO_p+$$
$$\sum_{a\in NR}\sum_{b\in N_a} PRI\cdot WPIO_{a,b}+$$
$$\sum_{m\in MR} PRS\cdot QMIO_m+$$
$$\sum_{a\in NR}\sum_{b\in N_a} PRI\cdot WMIO_{a,b}+$$
$$\sum_{a\in NR} PO \quad (1)$$

Equation (1) gives the MILP model objective, which is to minimize the resource provisioning power consumption and the communication fabric power consumption, where XP_j^p is a binary indicator and XP_j^p =1 indicates that processor j in CPU rack p is active; otherwise, XP_j^p = 0. XM_j^m, and XIO_j^{io} are defined in a similar manner for the memory and IO respectively.

ΔP=Pmax-Pmin, where ΔP is referred to as the CPU power factor (Watt), Pmax and Pmin are the CPU maximum and idle (i.e. minimum) powers respectively (Watt). ΔM and ΔIO are the memory and IO power factors respectively. δP_j^p is the total utilization of processor j in CPU rack p (unitless). δM_j^m and δIO_j^{io} are the memory and IO total utilization respectively. PRS and PRI are the electrical switch port power for the source and intermediate nodes respectively (Watt). PRS and PRI will be explained later in detail in order to show the difference between their values. QPM_p and $QPIO_p$ are the number of aggregation ports of the MEXC and IOEXC electrical switches in CPU rack p respectively, and $QMIO_m$ is the number of aggregation ports of the electrical switch at memory rack m.

$WPM_{a,b}$, $WPIO_{a,b}$ and $WMIO_{a,b}$ are the number of wavelengths that carry the CPU-MEM, CPU-IO and MEM-IO traffic in the physical link (a,b) respectively.

For simplicity and due to their small power consumption, we assume that the optical switches are always on. Note that NR unites all of PR, MR, and IOR, and therefore the optical switch power PO (Watts) is summed once over NR.

The minimization is subject to:

1) Resource Allocation Constraints.

Capacity Constraints:

$$\delta P_j^p = \sum_{i\in VM}\theta P_{i,j}^p \le Utl \qquad \forall j\in NP,\ p\in PR \quad (2)$$

$$\sum_{p\in PR}\sum_{j\in NP}\theta P_{i,j}^p = \frac{VP_i}{PRO} \qquad \forall i\in VM \quad (3)$$

$$\theta P_{i,j}^p \le W\cdot YP_{i,j}^p \qquad \forall i\in VM, j\in NP, p\in PR \quad (4)$$

$$\theta P_{i,j}^p \ge e + YP_{i,j}^p - 1 \qquad \forall i\in VM, j\in NP, p\in PR \quad (5)$$

$$\delta M_j^m = \sum_{i\in VM}\theta M_{i,j}^m \le Utl \qquad \forall j\in NM, m\in MR \quad (6)$$

$$\sum_{m\in MR}\sum_{j\in NM}\theta M_{i,j}^m = \frac{VM_i}{MEM} \qquad \forall i\in VM \quad (7)$$

$$\theta M_{i,j}^m \le W\cdot YM_{i,j}^m \qquad \forall i\in VM, j\in NM, m\in MR \quad (8)$$

$$\theta M_{i,j}^m \ge e + YM_{i,j}^m - 1 \qquad \forall i\in VM, j\in NM, m\in MR \quad (9)$$

$$\delta IO_j^{io} = \sum_{i\in VM}\theta IO_{i,j}^{io} \le Utl \qquad \forall j\in NIO, io\in IOR \quad (10)$$

$$\sum_{io\in IOR}\sum_{j\in NIO}\theta IO_{i,j}^{io} = \frac{VIO_i}{IO} \qquad \forall i\in VM \quad (11)$$

$$\theta IO_{i,j}^{io} \le W\cdot YIO_{i,j}^{io} \qquad \forall i\in VM, j\in NIO, io\in IOR \quad (12)$$

$$\theta IO_{i,j}^{io} \ge e + YIO_{i,j}^{io} - 1 \qquad \forall i\in VM, j\in NIO, io\in IOR \quad (13)$$

Constraint (2) calculates the total processing utilization of each processor δP_j^p and ensures that it is less than the maximum allowed utilization (Utl). Constraint (3) calculates the utilization of each processor per allocated VM, $\theta P_{i,j}^p$, where VP_i is the processing demand of VM i and PRO is the CPU processing capacity (GHz). Constraints (4) and (5) allocate each VM to a certain processor in a certain CPU rack by evaluating $YP_{i,j}^p$, which is 1 if processor j in CPU rack p

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

hosts request i; otherwise, $YP_{i,j}^{p}$=0, using a very big number, W, and a very small number e, both numbers are used to help in the conversion to binary variables. Constraints (6)-(9) and (10)-(13) repeat the same steps of constraints (2)-(5), where the variables are defined in a similar manner but for the memory and IO modules respectively.

2) SLA Constraints:

$$\sum_{i\in VM} K_i \geq NVM \cdot SLA \quad (14)$$

$$K_{ip} \leq \sum_{p\in PR}\sum_{j\in NP} YP_{i,j}^{p} \quad \forall\, i \in VM \quad (15)$$

$$W \cdot K_{ip} \geq \sum_{p\in PR}\sum_{j\in NP} YP_{i,j}^{p} \quad \forall\, i \in VM \quad (16)$$

$$K_{im} \leq \sum_{m\in MR}\sum_{j\in NM} YM_{i,j}^{m} \quad \forall\, i \in VM \quad (17)$$

$$W \cdot K_{im} \geq \sum_{m\in MR}\sum_{j\in NM} YM_{i,j}^{m} \quad \forall\, i \in VM \quad (18)$$

$$K_{iio} \leq \sum_{io\in IOR}\sum_{j\in NIO} YIO_{i,j}^{io} \quad \forall\, i \in VM \quad (19)$$

$$W \cdot K_{iio} \geq \sum_{io\in IOR}\sum_{j\in NIO} YIO_{i,j}^{io} \quad \forall\, i \in VM \quad (20)$$

$$K_i = K_{ip} = K_{im} = K_{iio} \quad \forall\, i \in VM \quad (21)$$

Constraint (14) ensures that the total number of served VMs is within an acceptable predefined percentage of the incoming VMs requests according to the service level agreement (SLA) value. The total number of served VMs, K_i depends on the outcomes of constraints (15)-(21) collectively; if all these constraints yield a value of 1, then K_i is 1, otherwise K_i is 0, where K_{ip}, K_{im} and K_{iio} are binary variables indicating if the processing, memory and IO requirements of request i are, respectively, served or blocked.

3) Slicing Constraints:

$$\sum_{p\in PR}\sum_{j\in NP} YP_{i,j}^{p} \leq 1 \quad \forall\, i \in VM \quad (22)$$

$$\sum_{m\in MR}\sum_{j\in NM} YM_{i,j}^{m} \leq 1 \quad \forall\, i \in VM \quad (23)$$

$$\sum_{io\in IOR}\sum_{j\in NIO} YIO_{i,j}^{io} \leq 1 \quad \forall\, i \in VM \quad (24)$$

Constraint (22) ensures that a VM i processing requirement is served by only one CPU, i.e. this constraint prevents VM slicing. Constraints (23) and (24) repeat constraint (22) for the memory and IO requirements. If multiple VM copies or VM slicing is required, equations (22)-(24) should be upper bounded by an appropriate number greater than 1, however, for consistency with the CS design, we considered a scenario where each of the VM resource requirements in the DS design is also served by a single physical resource.

4) Active resources constraints:

Active processors

$$XP_j^p \leq W \cdot \delta P_j^p \quad \forall\, p \in PR, j \in NP \quad (25)$$

$$W \cdot XP_j^p \geq \delta P_j^p \quad \forall\, p \in PR, j \in NP \quad (26)$$

Active memory modules

$$XM_j^m \leq W \cdot \delta M_j^m \quad \forall\, m \in MR, j \in NM \quad (27)$$

$$W \cdot XM_j^m \geq \delta M_j^m \quad \forall\, m \in MR, j \in NM \quad (28)$$

Active IO modules

$$XIO_j^{io} \leq W \cdot \delta IO_j^{io} \quad \forall\, io \in IOR, j \in NIO \quad (29)$$

$$W \cdot XIO_j^{io} \geq \delta IO_j^{io} \quad \forall\, io \in IOR, j \in NIO \quad (30)$$

Constraints (25) and (26) jointly find the active processors by checking the utilization δP_j^p. Constraints (27) and (28) together check the active memory modules and constraints (29), and (30) repeat the same steps but for the IO modules.

5) Communication constraints

Generating the index matrix for the CPU-MEM traffic

$$PM_i^{p,m} \cdot 2 = \sum_{j\in NP} YP_{i,j}^{p} + \sum_{j\in NM} YM_{i,j}^{m} \quad \forall\, i \in VM, p \in PR, m \in MR \quad (31)$$

$$ZPM_i^{p,m} \leq PM_i^{p,m} \quad \forall\, i \in VM,\ p \in PR,\ m \in MR \quad (32)$$

$$ZPM_i^{p,m} \geq PM_i^{p,m} - 0.5 \quad \forall\, i \in VM,\ p \in PR,\ m \in MR \quad (33)$$

Generating the index matrix for the CPU-IO traffic

$$PIO_i^{p,io} \cdot 2 = \sum_{j\in NP} YP_{i,j}^{p} + \sum_{j\in Nio} YIO_{i,j}^{io} \quad \forall\, i \in VM, p \in PR, io \in IOR \quad (34)$$

$$ZPIO_i^{p,io} \leq PIO_i^{p,io} \quad \forall\, i \in VM,\ p \in PR,\ io \in IOR \quad (35)$$

$$ZPIO_i^{p,io} \geq PIO_i^{p,io} - 0.5 \quad \forall\, i \in VM,\ p \in PR,\ io \in IOR \quad (36)$$

Generating the index matrix for the Memory-IO traffic

$$MIO_i^{m,io} \cdot 2 = \sum_{j\in NP} YM_{i,j}^{m} + \sum_{j\in NIO} YIO_{i,j}^{io} \quad \forall\, i \in VM, m \in MR, io \in IOR \quad (37)$$

$$ZMIO_i^{m,io} \leq MIO_i^{m,io} \quad \forall\, i \in VM,\ m \in MR,\ io \in IOR \quad (38)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

$$ZMIO_i^{m,io} \geq MIO_i^{m,io} - 0.5 \quad \forall i \in VM,\ m \in MR,\ io \in IOR \quad (39)$$

Constraint (31) connects each source CPU rack to its destination memory rack. $PM_i^{p,m}$ is an integer indicator that connects the i^{th} VM CPU-MEM traffic to the relevant CPU and MEM racks. Constraints (32) and (33) collectively generate source-destination index matrix, $ZPM_i^{p,m}$ for all CPU-MEM traffic, depending on constraint (31). $ZPM_i^{p,m} = 1$ if the VM i generates traffic between CPU rack p and MEM rack m; otherwise, $ZPM_i^{p,m} = 0$. Constraints (34)-(36) and constraints (37)-(39) repeat the same steps of (31)-(33), where the variables are defined in a similar manner but for the CPU-IO traffic and Memory–IO traffic, respectively.

Generating the traffic demand matrix:

$$TPM_{p,m} = \sum_{i \in VM} VPM_i \cdot ZPM_i^{p,m} \quad \forall p \in PR,\ m \in MR \quad (40)$$

$$TPIO_{p,io} = \sum_{i \in VM} VPIO_i \cdot ZPIO_i^{p,io} \quad \forall p \in PR,\ io \in IOR \quad (41)$$

$$TMIO_{m,io} = \sum_{i \in VM} VMIO_i \cdot ZMIO_i^{m,io} \quad \forall m \in MR,\ io \in IOR \quad (42)$$

Constraint (40) generates the CPU-MEM traffic matrix $TPM_{p,m}$ based on the index matrix $ZPM_i^{p,m}$ calculated previously in constraints (32) and (33), and VPM_i, which is the i^{th} VM CPU-MEM traffic demands. Constraints (41) and (42) generate the CPU-IO and Memory-IO traffic matrices respectively, using the relevant variables.

Traffic flow conservation:

$$\sum_{b \in N_a} WPM_{a,b}^{p,m} - \sum_{b \in N_a} WPM_{b,a}^{p,m} = \begin{cases} (TPM_{p,m}\ /B) & a = p \\ -(TPM_{p,m}\ /B) & a = m \\ 0 & otherwise \end{cases} \quad \forall p \in PR, m \in MR, s \in NR \quad (43)$$

$$\sum_{b \in N_a} WPIO_{a,b}^{p,io} - \sum_{b \in N_a} WPIO_{b,a}^{p,io} = \begin{cases} (TPIO_{p,io}\ /B) & a = p \\ -(TPIO_{p,io}\ /B) & a = io \\ 0 & otherwise \end{cases} \quad \forall p \in PR, io \in IOR, s \in NR \quad (44)$$

$$\sum_{b \in N_a} WMIO_{a,b}^{m,io} - \sum_{b \in N_a} WMIO_{b,a}^{m,io} = \begin{cases} (TMIO_{m,io}\ /B) & a = m \\ -(TMIO_{m,io}\ /B) & a = io \\ 0 & otherwise \end{cases} \quad \forall m \in MR, io \in IOR, s \in NR \quad (45)$$

Constraints (43)-(45) are the flow conservation constraints for the CPU-MEM, CPU-IO, and Memory-IO traffic respectively in the networking elements switches, ensuring that the total incoming traffic is equal to the total outgoing traffic for all nodes except for the source and destination racks, where $WPM_{a,b}^{p,m}$, $WPM_{a,b}^{p,io}$ and $WPM_{a,b}^{m,io}$ are the number of wavelengths of lightpath (p,m), (p,io), and (m,io) respectively passing through a physical link (a,b).

Wavelengths capacity constraints:

$$\sum_{p \in PR} \sum_{m \in MR} WPM_{a,b}^{p,m} \leq WPM_{ab} \quad \forall a \in PR, b \in Na \quad (46)$$

$$\sum_{p \in PR} \sum_{io \in IOR} WPIO_{a,b}^{p,io} \leq WPIO_{ab} \quad \forall a \in PR, b \in Na \quad (47)$$

$$\sum_{m \in MR} \sum_{io \in IOR} WMIO_{a,b}^{m,io} \leq WMIO_{ab} \quad \forall a \in MR, b \in Na \quad (48)$$

Constraints (46)-(48) ensure that the summation of the number of wavelengths traversing a physical link in the optical layer does not exceed the total number of wavelengths in that link for the CPU-MEM, CPU-IO, and Memory-IO traffics respectively.

$$QPM_p = \frac{1}{B} \cdot \sum_{m \in MR} TPM_{p,m} \quad \forall p \in PR \quad (49)$$

$$QPIO_p = \frac{1}{B} \cdot \sum_{io \in IOR} TPIO_{p,io} \quad \forall p \in PR \quad (50)$$

$$QMIO_m = \frac{1}{B} \cdot \sum_{io \in IOR} TMIO_{m,io} \quad \forall m \in MR \quad (51)$$

Constraints (49)-(51) determine the total number of aggregation ports utilized by the CPU-MEM, CPU-IO, and Memory-IO traffics respectively in each rack by dividing the total traffic matrix by the wavelength rate B (Gbps).

B. Resource Provisioning in CS with Communication Power MILP Model

Here, we assume that both the CS and DS use the same data center networking topology, such as fat tree or spine and leaf. Therefore, the power consumption of this fabric is not included in both CS and DS designs, but we include the communication fabric needed to realize the DS functionality as shown in Fig. 3. Note that such topologies might not be an optimal choice for the DS data center, and a specially designed communication fabric on top of the communication layer shown in Fig. 3 may be needed to form an optimal higher networking architecture layer for the DS data center interconnect topology; this is an issue for future research.

In the CS MILP model we consider a pool of servers, rather than a pool of resources. Each CS is equipped with the same type of resources (CPU, memory, I/O) that are used in the DS. Furthermore, we assume that each CS has a single CPU associated with a single Memory and a single IO module. Therefore, the utilization of a certain resource in the CS will affect the utilization of the other two resources. For example CPU#1 is associated with memory#1 and IO module#1 in CS#1. If CPU#1 is fully utilized by VM#1 then memory#1 and IO module#1 cannot be used by another VM even if they

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

have enough capacity for the second VM. This is to be compared to the DS design. Thus, the number of servers in CS design is the same as the number of one type of resources, such as the number of CPU resources, and here we have 24 servers.

To account for the server power, we consider a fully loaded server power of 300 Watts [52]. The power consumption of each resource (e.g. CPU, memory, IO card) in this server was comparable to the values used in the DS. Given that all the traffic is contained within the same server, and we assume that VMs are not sliced among different servers and VMs do not interact with each other, we are not using switches for inter-server traffic with the CS design. Viewed differently, we essentially do not include the data centre communication fabric (eg. fat tree or spine and leaf) when comparing the CS and DS as the same fabric is considered in both cases and carries the same inter-server traffic, (although the DS can potentially benefit from an alternate custom higher layers architecture fabric as mentioned). Instead we account for the intra-server communication in the CS as follows: the power consumption of each resource was set exactly equal to the values used in DS to facilitate comparison, and the CS idle power consumption is considered to be 150W which is about 50% of its maximum power consumption [53][72], to account for the intra-server communication power. Later in the paper we revisit this idle power and consider the situation where only a fraction of it contributes to the intra-server communication power while the remaining non-communication power is distributed among other parts of the CS, e.g. non-IT components, hard disks, etc. Such non-communication power consumption has to be added to the DS design to accurately assess its power savings compared to the CS design.

For the CS approach, each VM is allocated to the server that has enough CPU, memory, and IO modules to accommodate the VM; otherwise, a new server is powered on to host the requesting VM.

In addition to the parameters and variables defined in Section
IV-A, we define the following:

Sets:

NS	Set of all servers
VM	Set of VMs to be served

Variables:

δP_j	Total processor utilization of server j
δM_j	Total memory utilization of server j
δIO_i	Total IO utilization of server j
X_j	Indicates if server j is active, $X_j = 1$; otherwise, $X_j = 0$
θP_{ij}	Portion of the processing capacity of server j allocated to request i
θM_{ij}	Portion of the memory capacity of server j allocated to request i
θIO_{ij}	Portion of the IO capacity of server j allocated to request i
Y_{ij}	$Y_{ij} = 1$ if server j hosts request i, otherwise, $Y_{ij} = 0$
NOS	Number of working servers

The resource provisioning in the CS-based data center MILP model is:

Objective: minimize:

$$\sum_{j\in NS}((X_j \cdot Pmin) + (\Delta P \cdot \delta P_j)) + \sum_{j\in NS}((X_j \cdot Mmin) + (\Delta M \cdot \delta M_j)) + \sum_{j\in NS}((X_j \cdot IOmin) + (\Delta \mathrm{IO} \cdot \delta IO_j)) + NOS \cdot 150 \quad (52)$$

The objective (52) aims to minimize the total power by consolidating VMs in the minimum number of working servers.

Capacity Constraints:

$$\delta P_j = \sum_{i\in VM} \theta P_{ij} \leq Utl \quad \forall j \in NS \quad (53)$$

$$P_j \cdot \theta P_{ij} = VP_i \cdot Y_{ij} \quad \forall i \in VM, j \in NS \quad (54)$$

$$\theta P_{ij} \leq W \cdot Y_{ij} \quad \forall i \in VM, j \in NS \quad (55)$$

$$\theta P_{ij} \geq e + Y_{ij} - 1 \quad \forall i \in VM, j \in NS \quad (56)$$

$$\delta M_j = \sum_{i\in VM} \theta M_{ij} \leq Utl \quad \forall j \in NS \quad (57)$$

$$M_j \cdot \theta M_{ij} = VM_i \cdot Y_{ij} \quad \forall i \in VM, j \in NS \quad (58)$$

$$\theta M_{ij} \leq W \cdot Y_{ij} \quad \forall i \in VM, j \in NS \quad (59)$$

$$\theta M_{ij} \geq e + Y_{ij} - 1 \quad \forall i \in VM, j \in NS \quad (60)$$

$$\delta IO_j = \sum_{i\in VM} \theta IO_{ij} \leq Utl \quad \forall j \in NS \quad (61)$$

$$IO_j \cdot \theta IO_{ij} = VIO_i \cdot Y_{ij} \quad \forall i \in VM, j \in NS \quad (62)$$

$$\theta IO_{ij} \leq W \cdot Y_{ij} \quad \forall i \in VM, j \in NS \quad (63)$$

$$\theta IO_{ij} \geq e + Y_{ij} - 1 \quad \forall i \in VM, j \in NS \quad (64)$$

Constraint (53) calculates the total processing utilization of each processor in each server and ensures that it is less than the maximum allowed utilization. Constraint (54) calculates the utilization of each processor per allocated VM, and constraints (55) and (56) allocate each VM to a certain processor in a certain server. Constraints (57)-(60) and (61)-(64) repeat the same steps of constraints (53)-(56) but for the memory and IO modules respectively.

Slicing Constraint:

$$\sum_{j\in NS} Y_{ij} = 1 \quad \forall i \in VM \quad (65)$$

Constraint (65) ensures that each VM will be served by one server. This constraint will force service quality equal to 100% SLA.

Active Resources Constraint:

$$X_j \leq W \cdot \delta P_j \quad \forall j \in NS \quad (66)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

$$W \cdot X_j \geq \delta P_j \qquad \forall j \in NS \qquad (67)$$

$$NOS = \sum_{j \in NS} X_j \qquad (68)$$

Constraints (66) and (67) find the working servers and constraint (68) uses their results to calculate the total number of working servers.

C. *Energy Efficient Resource Provisioning in Disaggregated Servers with Communication Fabric (EERP-DSCF) Heuristic*

The EERP-DSCF heuristic provides real-time implementation of the MILP model in Section IV-A. The flow chart of the heuristic is shown in Fig. 4. In this study, we use homogenous resources, and thus sorting resources according to their PF is not necessary. Therefore, the heuristic picks the first CPU from the first CPU rack in the cluster and uses it for serving the first VM request. Then, the heuristic decides the VM's memory and IO racks allocation based on joint criteria involving the resource availability and rack distance from the chosen CPU rack. Thus, both packing and open shortest path first (OSPF) algorithms are applied together.

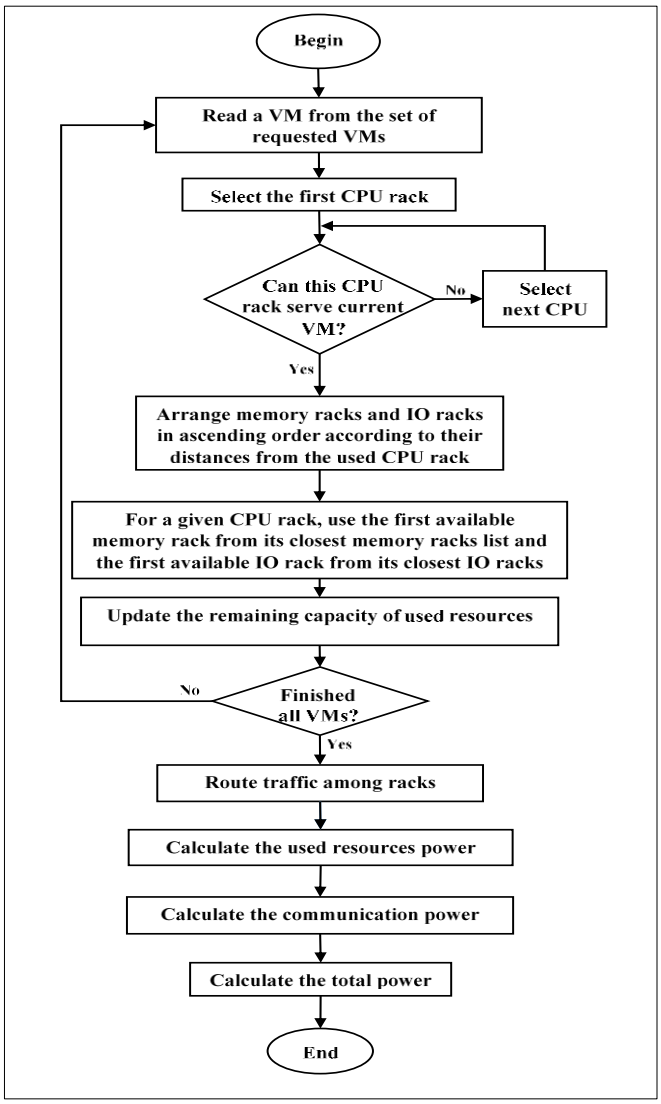


Fig. 4. EERP-DSCF heuristic flow chart.

As shown in the flow chart, Fig. 4, the heuristic follows a greedy approach to pack as many VMs as possible in the minimum number of resources and such resources may be physically far apart. Therefore, SLA violations might occur in the form of increased delay. However, delay constraints can be included by forming clusters of DS racks within a maximum distance limit between the selected racks in each cluster which should be set in consistency with the OSA metrics for server disaggregation.

The EERP-DSCF picks the first VM and allocates the first CPU in the first CPU rack in the cluster. The heuristic then organizes the memory and IO racks in a list according to their distances from the chosen CPU rack in ascending order. Subsequently, the heuristic checks the resources availability in the newly organized lists. If the first memory rack has enough capacity to accommodate the VM under consideration, the heuristic uses it; otherwise, the next rack is tested. In the same fashion, the heuristic checks the first IO rack in the list and allocates the chosen resources for the VM under consideration; otherwise, the next nearest IO rack is tested, and so on, until an available IO module is found. The heuristic tries to fill partially used resources and racks as much as possible before moving onto next racks. After allocating resources to the first VM request, the heuristic loops for the rest of the VM requests until all VMs allocations are done. Finally, EERP-DSCF grooms the traffic from each rack according to the traffic destination, routes them among racks, and calculates the total consumed power.

D. *Evaluation Scenarios for the MILP Model and Heuristic*

To evaluate the performance of the proposed MILP model and heuristic, we consider the example data center shown in Fig. 5. It consists of 72 racks (24 racks of each resource type), organized in an 8×9 matrix and 127 links, as shown in Fig. 5, which represents the top view of the DS data center considered.

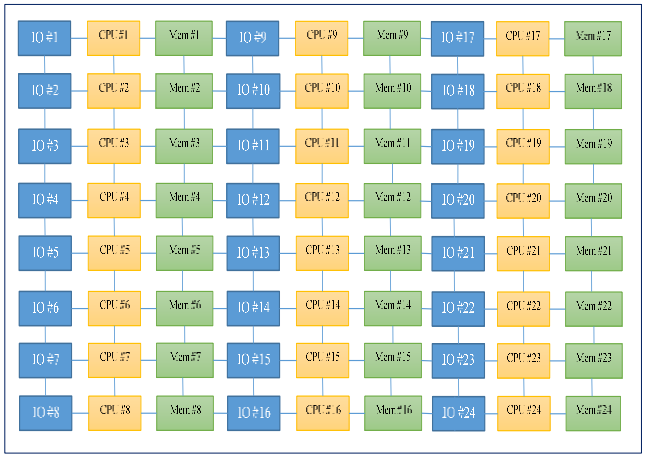


Fig. 5. DS based data center structure under consideration.

For the MILP model, a smaller version of the data center cluster, shown in Fig. 5, is used due to MILP computational complexity. It comprises 9 racks (3 racks of each resource type) organized in a 3×3 matrix and 12 links, as shown in Fig. 6. It follows the same structure and racks sequencing of the data center cluster in Fig. 5. The first column consists of three IO racks, the second consists of three CPU racks, and the last

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

column consists of three memory racks. We consider a scenario in which each rack contains 8 resources of its own type, each CPU rack contains 8 processors, each IO rack contains 8 IO modules, and each memory rack contains 8 memory modules. Thus every three racks, (one of each type, i.e. one processing, one memory and one IO racks) will contribute to make 8 servers. Therefore, in 3×3 racks there are 24 servers in total (8 × 3). Thus with a load of 25 VMs, assuming 1 VM per server, the maximum load of the DC is 100%.

For the heuristic, and due to its lower computational complexity, we evaluated the full 8×9 data center shown in Fig. 5. Each column is one type of resource racks, and each rack of each type contains 42 resources of its type. Starting from the far left, the first column contains the IO racks, followed by the CPU racks, then the Memory racks, and this sequence is repeated for the next 6 columns. Note that each rack is only connected to the nearest neighboring racks using optical fibers. As for the heuristic, the distance between adjacent racks is set to 1m [54].

For both the MILP model and the heuristic, each rack has its own intra-rack communication fabric and electrical and optical switches to facilitate the communication among racks and inter-rack communication, as shown in Fig. 1 and in Fig. 3. Table II shows the parameters used for both the MILP model and heuristic. The power consumption of the resources we used is consistent with our previous work in [33, 34], and, for the network devices, we use the values in Table II.

Regarding some of the power values given in Table II, it is worth noting that for the electrical switch port power and the packetizer power, we have linearly scaled up the values given in [55] for the switch port and [50] for the channel adapter to account for 100 Gbps port power. For the switch port power, and according to [55], a 10 Gbps port consumes 4 W; therefore, the switch port power consumption is calculated as 4×10W (equivalent to 100 Gbps port). For the 100 Gbps packetizer, in [50], the 40 Gbps packetizer power consumption is 7 W, and thus to scale up for 100 Gbps, the total power consumption is calculated as 7×2.5=17.5 W, and, to be more conservative, an extra 2.5 W was added, i.e. 20 Watts in total in order to account for other possible functionalities, such as buffering and arbitration.

TABLE II
INPUT PARAMETERS FOR THE MILP MODEL AND SIMULATION HEURISTIC

Power consumption of electrical switch port for source nodes *PRS*	70.5 W
Power consumption of electrical switch port for intermediate nodes *PRI*	43.5 W
Power consumption of electrical 100 Gbps switch port (Pr)	40 W [55]
Power consumption of an optical switch	85 W [56]
Bit rate of each wavelength	100 Gbps
CPU capacity	3.6 GHz [57]
CPU maximum power consumption	130 W [57]
RAM capacity	8 GB [57]
Memory maximum power draw	10.24 W [58]
IO module rate	10 Gbps [57]
IO module maximum power draw	21.4 W [59]
100 Gbps Optical transceiver power	3.5 W [49]
100 Gbps Multiplexer power (W)	4 W [51]
100 Gbps Packet engine (packetizer or packet engine) power	20 W [50]

We evaluate PRS by considering the scenario explained in Section IV-A, where a transceiver power is added to each end of each link and a packetizer power is considered for each source-destination pair.

$$PRS = 40\,W(\text{electrical switch port}) + 3{\times}3.5\,W\ (\text{transceivers power}) + 20\,W\ (\text{packetizer power})$$

Here, we consider 3 transceivers: the first is attached to the source resource, the second is for the destination resource, and the third is for the source electrical switch port.

For each intermediate node, we consider a switch port power plus a transceiver power, which yields:

$$PRI = 40\,W(\text{electrical switch port}) + 3.5\,W\ (\text{transceiver power})$$

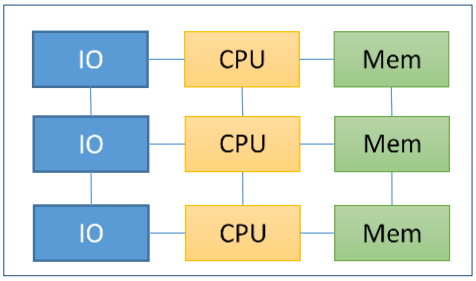


Fig. 6. Substrate data center for the MILP model.

E. VM Modeling

For the VM resources requirements and traffic demands, we use the requirements in Table III, which are assumed with respect to the three different VM types: processing intensive (PI), memory intensive (MI), and IO intensive (IOI). These three VM types are different in terms of their resources requirements, as shown in Table III. VMs are assumed to arrive at the same time, but we have not considered blocking because blocking can reduce power consumption, and we consider a situation where both CS and DS can accommodate all VM requests without blocking or migration to establish a comparison that is based solely on the difference in architecture and resource management between the two approaches. However, we note that VM blocking and different VM arrival patterns are interesting topics, and we leave their investigation for future research. Comparing the IO resources requirements to the actual traffic reveals that delay or maybe blocking situations can occur on the egress ports. However, we have not considered their effects in the analysis presented in this work.

A VM requires three resources: processing, memory and networking resources, and there is traffic between these resources. As mentioned above, we assume that all the VM requests arrive at once and they all have infinite service durations. Thus, it is a one shot (non-sequential) evaluation, such that a VM will use the same allocated resources at all times and will not be migrated to different resources. We assume all VMs are served as we set the SLA to 100% and as such no VM blocking is considered. We did not consider blocking VM requests since such blocking can lead to lower power consumption in one of the designs. Therefore we require both the CS and DS to serve all the requests.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

Using a computer with a 3.3 GHz CPU and 8 GB memory, our heuristic produced the results in less than one minute, considering 1000 IOI VMs. This is a remarkable improvement over the MILP model, which requires about 2 days to produce the results for only 20 IOI VMs using the same computer. While our heuristic results show the power consumption of a range of data center loads ranging from 100 to 1000 VMs, the MILP model shows results for data center loads ranging from 5 to 20 VMs, limited by the computational complexity of the MILP and the processing platform available.

TABLE III
INPUT PARAMETERS FOR THE VMS REQUIREMENTS

VM Type / Demands	PI	MI	IOI
CPU (GHz)	2-3.6	0.1-0.3	0.1-0.3
Mem (GB)	0.1-0.3	6-8	1-4
IO (Gbps)	0.5-1	0.5-1	6-10
CPU-MEM Traffic (Gbps)	10-100	10-50	5-20
CPU-IO Traffic (Gbps)	1-3	1-3	1-3
MEM-IO Traffic (Gbps)	1-3	1-5	6-10

F. *MILP Model and Heuristic Results*

Fig.7 compares the average power consumption results of the MILP model for the DS and CS designs and the DS heuristic considering the three VM types PI, MI and IOI. The results were obtained by running the MILP models and the heuristic for the cases of 5, 10, 15, 20 and 25 VM requests, then the average power consumption is calculated. The results show clearly that the DS heuristic and the DS MILP are comparable and the heuristic follows the MILP closely

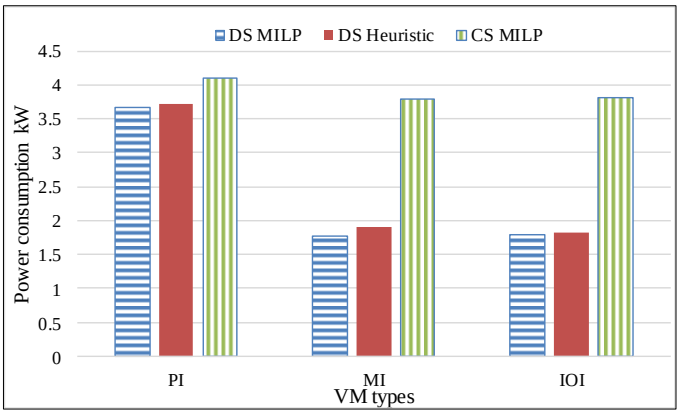


Fig. 7. Average power consumption comparison of the DS MILP model, DS heuristic, and CS MILP model with communication fabric.

Examining the results in Fig. 7 and comparing the DS MILP to the CS MILP shows that the PI VMs are the highest power consuming demands which leads to minimum average power saving, of about 3% while MI and IOI have comparable power consumptions and they have comparable average power savings, of about 42%.

Fig. 7 shows that the heuristic and MILP are close and the heuristic execution time is short. Therefore, we have obtained new results where for each VM the CPU load was random and uniform between 0.1 and 3.6 GHz, the other VM attributes were also random and uniform and were: Mem (1-8) GB, IO (0.5-10) Gbps, CPU-Mem traffic (10-100) Gbps, CPU-IO traffic (1-3) Gbps, Mem-IO traffic (1-10) Gbps.

The power savings are averaged over 100 experiments (each experiment obtained power saving using 100 runs of the CS and DS heuristics for 5, 10, 15, 20, and 25 VMs) and the power profiles are shown in Fig. 8. With the considered high demand values, the DS power consumption is more than the CS for very small numbers of VMs, but moving to large numbers of VMs, the DS shows better performance compared to CS due to good resource packing.

The placement of VMs is optimized by the CS and the DS heuristics so as to minimize the power consumption through packing and proximity placement as shown in Fig. 4.

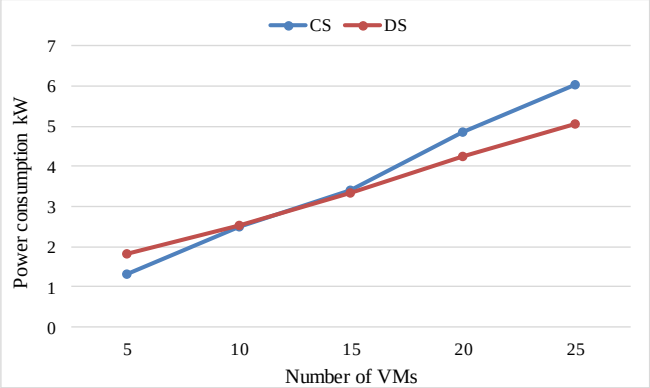


Fig. 8. Average power consumption for different numbers of VMs considering 100 runs for each case.

Fig. 9 shows the DS heuristic results for the power consumption of the networking resources (NetP), the resource power consumption (RESP), and the total power consumption (Total). The results can be explained by considering the cluster topology and size under consideration plus the number of served VMs, as well as the inputs, in particular, the resource specifications, and VM requirements.

Fig. 9.a shows the PI VM requests results. Note that the resource power is higher than the networking power, and it has higher impact on the total power consumption. Given the parameters in Tables II and III for the resources, power consumption, and VM demand values, the average CPU demand per VM is 3 GHz for the PI VMs type, and thus a huge number of CPUs will be used for VM allocation. A lot of VMs will occupy a whole CPU on their own and not share the CPU with other VMs. Given that the CPU has high power consumption values, the resource power consumption is the highest and exceeds the network power consumption.

In relation to the networking power, Fig. 10 shows the active racks of each type when serving 1000 VM requests for the three VM types, PI, MI, and IOI. Examining Fig. 10 reveals that considering 1000 PI VM requests results in a case where all the racks in the cluster are activated, regardless of their utilization, where an active rack means there is an outgoing/incoming traffic. Each CPU rack has an active memory and IO racks among its neighbors, enforced by the heuristic, and thus all traffic, in this case, will be a single hop traffic resulting in about 80 kW networking power consumption, which is less than the power consumed by the resources.

Fig. 9.b shows the evaluation for the MI VM requests and compares the two power components: the network and resource powers. As in PI, both the network power and resource power consumption increase with increasing number

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

of VMs. The increase in resource power is far less, however, than the increase in network power consumption, as memories consume few watts, and only two CPU racks are enough to accommodate the processing requirements of all the VMs under consideration. Inspecting Fig. 10, and the MI results in particular, all the memory racks are used, but only two CPU racks and two IO racks are used. This is due to the approach followed by our heuristic when performing the resource allocation and packing. The heuristic first allocates the best available CPU to reduce the number of working CPUs, resulting in only two working CPU racks, whereas the memory and IO allocations follow the CPU allocation by choosing the closest memory and IO racks to the used CPU rack that have enough capacity to accommodate the VM under consideration.

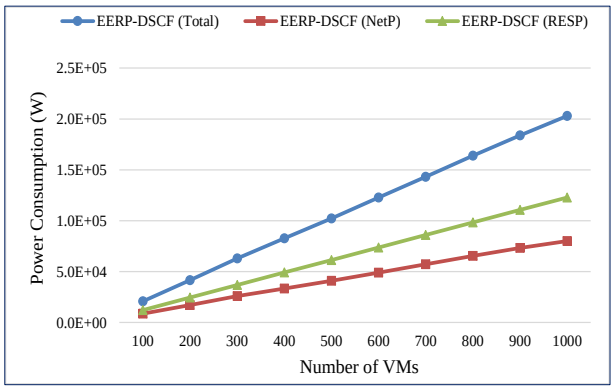
Thus, the traffic due to these CPU racks destined to the memory racks, which typically has moderate values, has to travel through long pathways, passing a significant number of multi-hop links and further increasing the network power consumption. In the same manner, the traffic from the memory racks to the IO racks traverses almost the whole cluster, in some cases, to reach its destination. This leads to about 138 kW networking power consumption, following resource packing, which reduces the number of active resources (processing resources especially), resulting in the minimum resources power consumption.

Fig. 9.c shows the power consumption of the DS heuristic for IOI VMs. As the IOI VM type has the lowest CPU-MEM traffic values, and due to both good resource packing for this energy efficient module and traffic routing by the heuristic, this scenario resulted in the minimum network power consumption and minimum resources power consumption which yielded minimum total power consumption. Fig. 10 reveals that all IO racks are working in addition to only two power consuming CPU racks and seven memory racks. Again, this can be explained by observing the way the heuristic works. The heuristic's first priority is to allocate the VMs in the smallest number of CPUs. After that, the memory and IO modules allocation follows the CPU allocation by choosing the closest available resources to the used CPUs.

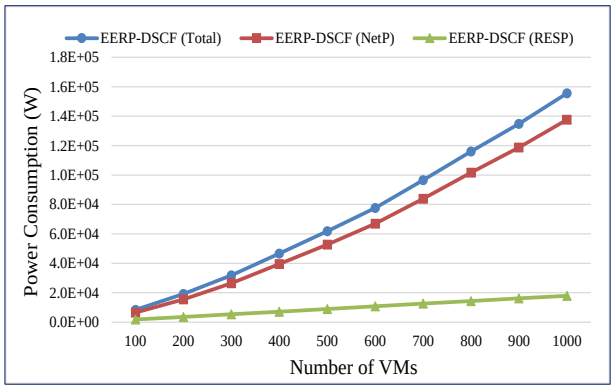
It can clearly be seen in Fig. 10 that the heuristic preferred memory racks # 9 and #10 instead of #6, #7 and #8. Examining Fig. 5 shows that CPU racks 1 and 2 are closer to memory racks 9 and 10 compared to racks 6, 7 and 8.

To show the effect of the power consumption attributed to communications on the overall power saving, Fig. 11 compares the average power consumption of the CS based data center design to the power consumption of the DS based data center design for a large number of VMs ranging from 100 to 1000 VMs and considering the PI, MI, and IOI VM types. The CS approach is implemented as a heuristic, where the total number of coupled resources required to form server units to serve incoming VMs are determined, and then 150 W is added to the power consumption of the resources of each active server to account for the internal communication overhead. Fig. 11 shows that the average power saving for the processing intensive DS with communication fabric (PI DSCF) is about 10% when compared with the processing intensive CS with communication fabric (PI CSCF). This is due to the use of the power-hungry processing resources in both DS and CS designs to a high extent when compared to the number of used memory and IO resources in the DS.

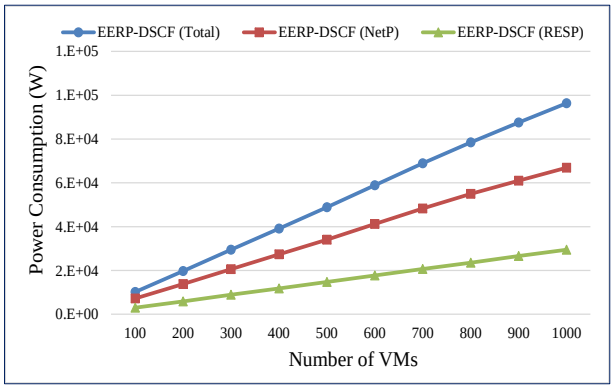
However, due to the DS ability to pack a higher number of VMs in fewer resources (i.e. memory and IO in this case), DS managed to save a considerable fraction of the power compared to CS. Regarding the networking power consumption, as discussed earlier in this section, the allocation of VMs in the DS racks led to a communication pattern such that all traffic passes single hop paths, leading to moderate network power consumption in spite of having high traffic values associated with the PI VM demands, and leading to overall total power saving compared to CS.



(a) PI



(b) MI



(c) IOI

Fig. 9. The power consumption of the EERP-DSCF with a large number of VMs.

Fig. 11 shows the average power saving for the memory intensive DS with communication fabric (MI DSCF) design. The saving is about 53% compared to the memory intensive

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

CS with communication fabric (MI CSCF). The higher saving percentage for the MI results when compared with the previous PI results is due to the efficient utilization of the power intensive CPU resources in the DS compared to the CS. In the CS a large number of servers has to be powered-on due to the congestion on the memory modules.

In relation to the networking power consumption, serving MI requests increases the network power consumption in DS when compared with PI and IOI demands as shown in Fig. 9. Having a large number of working memory racks and a lower number of working CPU and IO racks increases the traffic among these racks and many traffic flows take multi-hop paths. However, the DS total power consumption (resources plus networking) is still lower than the CS total power consumption, as CS operates a large number of servers, roughly the same as the number of powered-on memory modules in the DS, which maintain the higher CS power consumption when compared to DS.

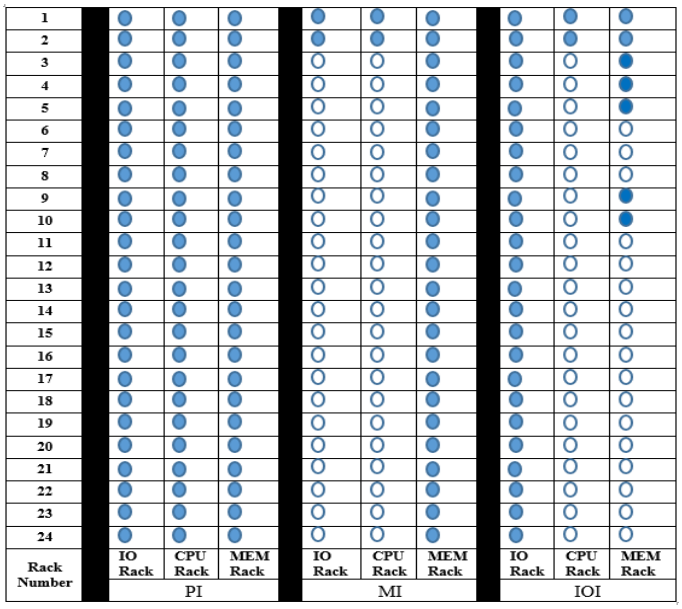


Fig. 10. Active racks considering 1000 VM requests.

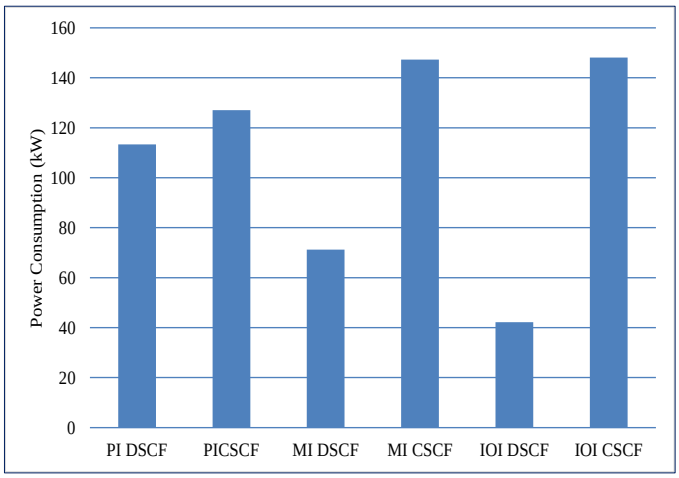


Fig. 11. Average power consumption of EERP-DSCF compared to CS with communication fabric.

Similarly, the IOI VMs scenario resulted in the highest power saving, with an average power saving of about 63% compared to CS.

For the resource power, the IOI VMs consume less power than the PI VMs but more than the MI VMs scenarios. According to Fig. 10, the communication power in IOI VMs, as explained earlier, is much lower than the MI VMs scenario, thus, the IOI VM scenario total power is the smallest among other VM types, resulting in the highest power saving.

Finally, we investigate the other extreme scenarios represented by mixed VM demands, such as PI+MI VMs or PI+MI+IOI VMs. Table IV captures the requirements of these mixed VM types. Note that we selected the maximum demand values in each set of VM combinations to establish the power savings limits.

TABLE IV
INPUT MIXED VMS RESOURCES AND TRAFFIC REQUIREMENTS

Demands \ VM Type	PI+MI	PI+IOI	MI+IOI	PI+MI+IOI
CPU (GHz)	2-3.6	2-3.6	0.1-0.3	2-3.6
Mem (GB)	6-8	1-4	6-8	6-8
IO (Gbps)	0.5-1	6-10	6-10	6-10
CPU-MEM Traffic (Gbps)	10-100	10-100	10-50	10-100
CPU-IO Traffic (Gbps)	1-3	1-3	1-3	1-3
MEM-IO Traffic (Gbps)	1-5	6-10	6-10	6-10

The VMs requirements, resources, and traffic values have been chosen to cover the extreme values for the considered types to cover a variety of VMs categories. Fig. 12 shows the total power consumption of the four different scenarios mentioned in Table IV, for both DS and CS servers.

The MI+IOI and PI+MI scenarios have the highest power savings, about 28% and 27% respectively. For the first scenario (MI+IOI), due to the low CPU demands and relatively low CPU-MEM traffic values compared to the other VM types, this scenario resulted in the best power profile followed by the PI+MI scenario.

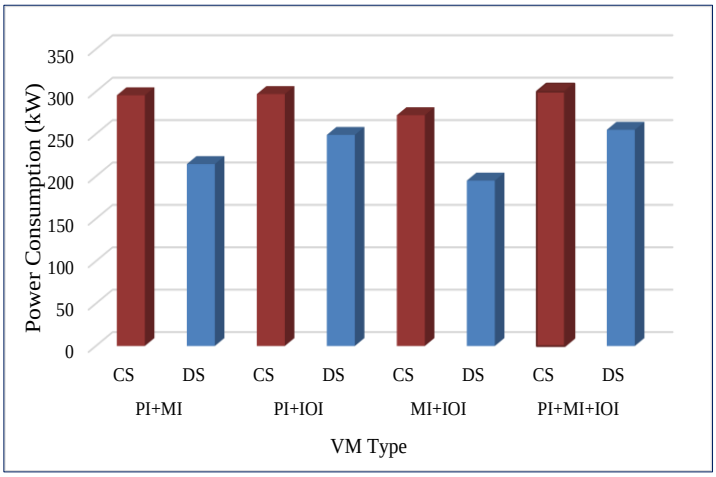


Fig.12. The power consumption of DS and CS heuristics considering a variety of 1000 VM requests.

The mixed PI+MI scenario has a higher CPU demand compared to the MI+IOI VMs scenario, but lower IO demand and MEM-IO traffic compared to the other scenarios (PI+IOI and PI+IOI+MI). Thus, it has higher power savings than both, but lower than MI+IOI due to its high CPU demand and CPU-MEM traffic. The last two scenarios, PI+IOI and PI+MI+IOI,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

resulted in the minimum power savings: 17%, and 15%, respectively. This is consistent with the resources demands and traffic values required by these VM types. The worst scenario PI+MI+IOI has all the highest values thus it has the worst saving while the PI+IOI has also all the highest values except for the least power consumers, the memory demands, which resulted in low power saving but slightly more than the PI+MI+IOI.

Regarding the traffic values, Fig. 13 is a case study which shows the effect of having high CPU-MEM traffic on the total power saving considering 1000 IOI VMs. This case study investigates the highest power saving scenario, IOI VMs (note that IOI VMs represent video streaming which currently accounts for about 90% of the traffic in the Internet, ie outside the data center [60]. The case study clearly shows that increasing the high CPU-MEM traffic increases the DS power consumption until it reaches the point where both designs have the same power profiles at around 225 Gbps traffic rate.

With the increase in the traffic, the CS design gives a better power profile than the DS design. However, the evaluations conducted in this paper are based on current technologies with very conservative assumptions. With future improved communications technologies, the DS architecture is expected to be a promising choice over several dimensions, especially the energy-saving dimension.

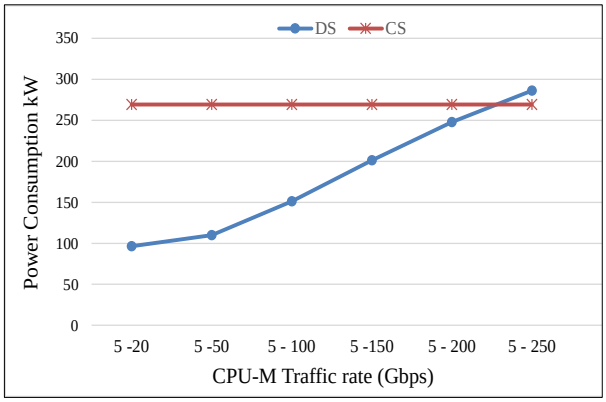
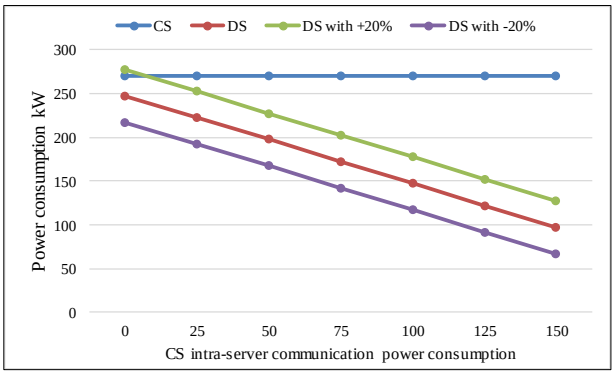
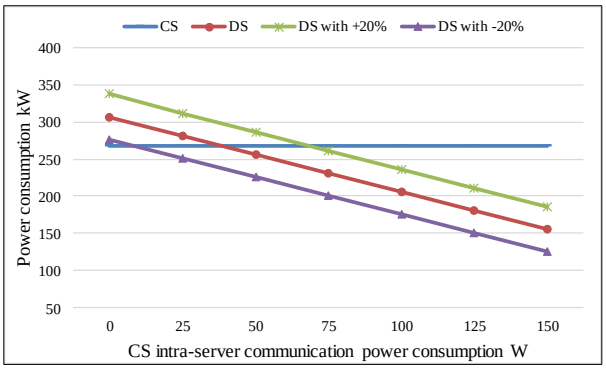


Fig. 13. Heuristic results showing the effects of increasing the CPU-MEM traffic on the total power consumption considering 1000 IOI VMs.

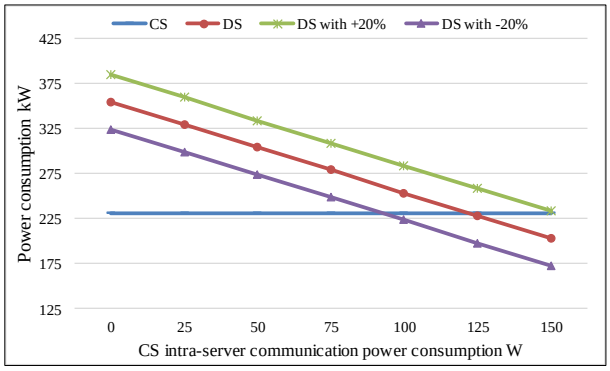
The results in this section so far computed the DS power as the sum of the power consumption of CPU, IO and memory. Therefore the results were based on a 150W idle power consumption that is all attributed effectively to communications in the CS. A fairer comparison is presented in Fig. 14 (which can be compared to the 1000 VMs case in Fig. 8) where the proportion of idle power attributed to communications in the CS is varied from zero up to the maximum idle power of 150W. Note that any power attributed to communications in the CS is not moved to the DS. The remaining part of the idle power (not attributed to communications) is moved to the DS and may account for hard disks, fans etc. Furthermore, we report sensitivity analysis where the non-IT power consumption (cooling, power supply, etc.) in the DS design was changed by ± 20%. The increase in non-IT power consumption indicates a less efficient cooling and power supply system for the rack scale DS design. The decrease can mean that sharing cooling and using large centralized power supplies may result in non-IT power savings in the DS design. Therefore, our results explore a range of possible future evolution scenarios in areas (cooling and power supply) that are beyond the scope of communication, networking and lightwave design, the focus of this paper.



(a) IOI



(b) MI



(c) PI

Fig. 14. Heuristic results showing the impact of considering different non-communication power consumption values in DS, under 1000 IOI, MI and PI VMs.

As can be seen in Fig.14, serving IOI VMs, can still produce considerable power savings with an average of 36% power saving . Serving MI VMs has an average power saving of 13% while the least power saving scenario (serving PI VMs) has savings only when the CS communication power is remarkably high (125 W or more), or when the DS fabric power consumption decreases with progress in communications and switching power consumption reduction.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JLT.2017.2767574, Journal of Lightwave Technology

V. Disaggregated Server Implementation Technologies and Limitations

In our new design, packets have special packet formats. The CPUMC encapsulates the memory address and control information, such as read, write, number of successive bytes, and any other information, as an Ethernet packet, for communication between the processor and memory modules located in different racks. The IO packet engine encapsulates the IO address and control information for the CPU IO communication, as an Ethernet packet. For example, a packet sent from a CPU contains an address part (header) and data (payload). The address contains the IP of the destination rack and the ID of the specific module memory or IO which the CPU wants to access. These are provided by the data center global operating system. The rack IP address is used by the CPU rack MEXC or IOEXC switch to forward the packet to the destination rack. On receiving the packet at the destination rack, the top of rack switch, MMC or IOEXC, reads the specific module ID and forwards the packet to its right destination module. Thus, all the communication passes through the electronic (Ethernet) ToR switches. For high performance computing, such as the DS data center, low latency switching is a key element to enable upper layer latency sensitive applications; thus, switch latency is becoming a very critical factor [27].

Here, we describe some proposals that can further reduce the total latency in our design: (i) we suggest the use of a reduced switching protocol overhead and simple packet format due to the topology and data nature, which will jointly help in reducing the total system latency; (ii) our switches could be designed specifically for certain packets formats, like the CPU-MEM, CPU-IO, and MEM-IO, instead of generic IP switches; (iii) we propose the use of flexible protocol formats to handle different applications that have different latency restrictions. Thus, for latency tolerant applications, we allow the use of implicit circuit switching by establishing dedicated channels for a given time for this specific application; (iv) the use of MPLS as a simple switching technique or implicit circuit switching with time division multiplexing (TDM); (v) implementing optical switching (circuit [61], packet [62], [63], [64], label [63], and burst switching [65]) as a fast and reliable switching technique, which will also eliminate the need for some of the optical transceivers which perform optical to electrical to optical conversion when electronic switches are used. For example, optical burst switching can be useful for memory communications which are typically bursty in nature (e.g. file downloading). The elimination of the packetizer/depacketizer is also attractive; (vi) finally, by looking at the latency reduction trends in recent years in Ethernet switches [66], [67], [68]; attributed to new advanced switching architecture design and improved silicon technology, the Ethernet switch latency is decreasing from double-digit milliseconds to sub-microsecond [69]. With this trend, it is highly expected that the Ethernet switch latency will arrive at a point that fits the DS requirements.

For the memory modules, we propose the use of high performance components to overcome latency and communication delay bottlenecks. DDR4 [70] is the latest version of RAM technology, offering a range of improvements over its predecessor, DDR3 [71], such as greater range of available clock speeds and timings and lower power consumption.

Compared to OSA recommendation for a 50ns round trip delay between the CPU and the memory [27], the current switching technology needs to further improve to comply with such requirements in the introduced DS design. Considering some of the previously mentioned implementation ideas and future technology development, this design can be a promising implementation for the DS. We believe that the switching times of optical and electrical switches will continue to improve with time.

VI. Conclusions and Future Work

This paper has investigated the advantages of disaggregated server (DS) design over traditional monolithic conventional server (CS) design. First, we presented our new design for the photonic-based DS data center architecture, supplemented with a complete description of the architecture components and communication patterns. Second, we analyzed the energy efficient resource provisioning and VM allocation in DS server design with communication fabric. A MILP optimization was developed for the purpose of optimizing VM allocation for DS-based data centers, considering the communication fabric power consumption. The results show that considering pooled resources yields considerable power savings when compared with the CS approach, and up to 42% average power savings were achieved based on the MILP optimized system. Third, for real-time implementation, we developed an energy efficient resource provisioning heuristic for DS (EERP-DSCF), based on the MILP model insights, with comparable power efficiency to the MILP model. The effect of CPU-MEM traffic values has been investigated by increasing the CPU to memory traffic and comparing the DS and CS power consumption. In addition, the impact of the inclusion of non communication power has been considered by adding an extra 150 W non communication power (per CS equivalent server) to the DS design. Finally, some recommendations for design and implementation focus on the requirements, the capabilities of different switching and implantation technologies, and the challenges that this architecture can face. Planned future work includes consideration of virtualization in DS-based data centers, geo- distributed DS-based data centers, VM scheduling, real-time VM migration, and VM blocking with appropriate MILP models and heuristics together with consideration of other performance metrics (beyond power consumption) such as scalability, resilience, latency and security.

References

[1] TechTarget, "A disaggregated server proves breaking up can be a good thing", SearchDataCenter, 2016. [Online]. Available: http://searchdatacenter.techtarget.com/feature/A-disaggregated-server-proves-breaking-up-can-be-a-good-thing. [Accessed: 01- Dec- 2016]."

[2] J. Baliga, R. W. Ayre, K. Hinton, and R. S. Tucker, "Green cloud computing: Balancing energy in processing, storage,

and transport," *Proceedings of the IEEE,* vol. 99, pp. 149-167, 2011.

[3] "L. Barroso and U. Hölzle The case for energy-proportional computing computer," *IEEE Computer*, vol. 40, pp. 33-37, 2007.

[4] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *ACM SIGCOMM Computer Communication Review*, 2008, pp. 63-74.

[5] A. Hammadi, T. El-Gorashi, and J. Elmirghani, "Energy-Efficient Software-Defined AWGR-Based PON Data Center Network," 2016.

[6] B. Kantarci, L. Foschini, A. Corradi, and H. T. Mouftah, "Inter-and-intra data center VM-placement for energy-efficient large-scale cloud systems," in *2012 IEEE Globecom Workshops*, 2012, pp. 708-713.

[7] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM computer communication review*, vol. 39, pp. 68-73, 2008.

[8] X. Dong, T. El-Gorashi, and J. M. Elmirghani, "Green IP over WDM networks with data centers," *Journal of Lightwave Technology*, vol. 29, pp. 1861-1880, 2011.

[9] A. Q. Lawey, T. E. El-Gorashi, and J. M. Elmirghani, "Distributed energy efficient clouds over core networks," *Journal of Lightwave Technology*, vol. 32, pp. 1261-1281, 2014.

[10] A. Q. Lawey, T. E. El-Gorashi, and J. M. Elmirghani, "BitTorrent content distribution in optical networks," *Journal of Lightwave Technology*, vol. 32, pp. 3607-3623, 2014.

[11] X. Dong, T. E. El-Gorashi, and J. M. Elmirghani, "On the energy efficiency of physical topology design for IP over WDM networks," *Journal of Lightwave Technology*, vol. 30, pp. 1694-1705, 2012.

[12] X. Dong, T. El-Gorashi, and J. M. Elmirghani, "IP over WDM networks employing renewable energy sources," *Lightwave Technology, Journal of*, vol. 29, pp. 3-14, 2011.

[13] L. Nonde, T. E. El-Gorashi, and J. M. Elmirghani, "Energy efficient virtual network embedding for cloud networks," *Journal of Lightwave Technology*, vol. 33, pp. 1828-1849, 2015.

[14] A. Pagès, J. Perelló, F. Agraz, and S. Spadaro, "Optimal VDC Service Provisioning in Optically Interconnected Disaggregated Data Centers," *IEEE Communications Letters*, vol. 20, pp. 1353-1356, 2016.

[15] I. "The Case for Rack Scale Architecture", 2016. [Online]. Available: http://www.intel.com/content/www/us/en/architecture-and-technology/rsa-introduction-paper.html. [Accessed: 01-Dec- 2016].

[16] I. "Design Guide for Photonic Architecture", 2016 [Online]. Available: http://opencompute.org/assets/Uploads/Open_Compute_Project_Open_Rack_Optical_Interconnect_Design_Guide_v0.5.pdf. [Accessed: 01- Dec- 2016].

[17] Intel, "New Photonic Architecture Promises to Dramatically Change Next Decade of Disaggregated Rack Scale Server Designs". [Online]. Available: https://newsroom.intel.com/news-releases/intel-facebook-collaborate-on-future-data-center-rack-technologies. [Accessed: 01- Dec- 2016]."

[18] P. Grun, "Introduction to infiniband for end users". [Online]. Available: http://www.mellanox.com/pdf/whitepapers/Intro_to_IB_for_End_Users.pdf. [Accessed: 10-Jun-2017]," *White paper, InfiniBand Trade Association,* 2010.

[19] K. Lim, Y. Turner, J. R. Santos, A. AuYoung, J. Chang, P. Ranganathan, *et al.*, "System-level implications of disaggregated memory," in *IEEE International Symposium on High-Performance Comp Architecture*, 2012, pp. 1-12.

[20] K. Lim, Y. Turner, J. Chang, J. R. Santos, and P. Ranganathan, "Disaggregated Memory Benefits for Server Consolidation"." *HP Laboratories*, 2011.

[21] K. Lim, J. Chang, T. Mudge, P. Ranganathan, S. K. Reinhardt, and T. F. Wenisch, "Disaggregated memory for expansion and sharing in blade servers," in *ACM SIGARCH Computer Architecture News*, 2009, pp. 267-278.

[22] K. T.-M. Lim, "Disaggregated memory architectures for blade servers". [Online]. Availabe: http://web.eecs.umich.edu/~tnm/trev_test/dissertationsPDF/kevinL.pdf. [Accessed: 10-Jun-2017]," Hewlett-Packard Labs, 2010.

[23] S. Han, N. Egi, A. Panda, S. Ratnasamy, G. Shi, and S. Shenker, "Network support for resource disaggregation in next-generation datacenters," in *Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks*, 2013, p. 10.

[24] P. Svärd, B. Hudzia, J. Tordsson, and E. Elmroth, "Hecatonchire: enabling multi-host virtual machines by resource aggregation and pooling," *Digitala Vetenskaplika Arkivet*, 2014.

[25] C.-C. Tu, C.-t. Lee, and T.-c. Chiueh, "Marlin: a memory-based rack area network," in *Proceedings of the tenth ACM/IEEE symposium on Architectures for networking and communications systems*, 2014, pp. 125-136.

[26] Cisco, "Cisco Composable Infrastructure Optimize Your Infrastructure for Each Application in Seconds", Cisco, 2016 [Online]. Available: https://www.cisco.com/c/dam/en/us/products/collateral/servers-unified-computing/ucs-m-series-modular-servers/at-a-glance-c45-735889.pdf. [Accessed: 01- Dec- 2016]."

[27] OSA, "Photonics for Disaggregated Data Centers Workshop," 2015.

[28] Y. Yan, Y. Shu, G. M. Saridis, B. R. Rofoee, G. Zervas, and D. Simeonidou, "FPGA-based optical programmable switch and interface card for disaggregated OPS/OCS data centre networks," in *Optical Communication (ECOC), 2015 European Conference on*, 2015, pp. 1-3.

[29] Y. Yan, G. M. Saridis, Y. Shu, B. R. Rofoee, S. Yan, M. Arslan, *et al.*, "All-Optical Programmable Disaggregated Data Centre Network Realized by FPGA-Based Switch and Interface Card," *Journal of Lightwave Technology*, vol. 34, pp. 1925-1932, 2016.

[30] Intel and Tancent, "Tencent Explores Datacenter Resource Pooling Using Intel® Rack Scale Architecture (Intel® RSA)". [Online]. Available: http://www.intel.com/content/dam/www/public/us/en/documents/white-papers/rsa-tencent-paper.pdf. [Accessed: 01-Dec- 2016]."

[31] B. Abali, R. J. Eickemeyer, H. Franke, C.-S. Li, and M. A. Taubenblatt, "Disaggregated and optically interconnected memory: when will it be cost effective?," *arXiv preprint arXiv:1503.01416*, 2015.

[32] K. Katrinis, D. Syrivelis, D. Pnevmatikatos, G. Zervas, D. Theodoropoulos, I. Koutsopoulos, *et al.*, "Rack-scale disaggregated cloud data centers: The dReDBox project vision," in *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2016, pp. 690-695.

[33] H. M. M. Ali, A. Q. Lawey, T. E. El-Gorashi, and J. M. Elmirghani, "Energy efficient disaggregated servers for future data centers," in *Networks and Optical*

Communications, 2015. NOC 20th European Conference on, 2015, pp. 1-6.

[34] H. M. Ali, A. Lawey, T. E. Elgorashi, and J. Elmirghani, "Energy Efficient Resource Provisioning in Disaggregated Data Centres," in *Asia Communications and Photonics Conference*, 2015, p. AM1H. 1.

[35] H. Mohammad Ali, A. Al-Salim, A. Q. Lawey, T. El-Gorashi, and J. M. Elmirghani, "Energy Efficient Resource Provisioning with VM Migration Heuristic for Disaggregated Server Design," 2016.

[36] B. Akesson, "An introduction to SDRAM and memory controllers". [Online]. Available: http://www.es.ele.tue.nl/~premadona/files/akesson01.pdf. [Accessed: 01- Dec- 2016]."

[37] J. P. Hayes, *Computer architecture and organization*: McGraw-Hill, Inc., 2002.

[38] H. M. Servers. *Energy Efficient Integrated Infrastructure Servers". [Online]. Available: https://www.hpe.com/uk/en/servers/moonshot.html. [Accessed: 01- Dec- 2016]*.

[39] IBM, "Why Blade Servers?. [Online]. Available: http://www-05.ibm.com/hu/termekismertetok/xseries/dn/why_blade_servers.pdf [Accessed: 16- Oct- 2017]."

[40] Y. Zhang and N. Ansari, "HERO: Hierarchical energy optimization for data center networks," *IEEE Systems Journal*, vol. 9, pp. 406-415, 2015.

[41] P. Costa, "Bridging the gap between applications and networks in data centers," *ACM SIGOPS Operating Systems Review*, vol. 47, pp. 3-8, 2013.

[42] QuickLogic, "Dual-processor Architectures for Smartphones and Tablets".[Online]. Available: http://www.quicklogic.com/technologies/connectivity/ipc/. [Accessed: 20- Feb- 2017]."

[43] J. Xu, M. Zhao, J. Fortes, R. Carpenter, and M. Yousif, "On the use of fuzzy modeling in virtualized data center management," in *Fourth International Conference on Autonomic Computing (ICAC'07)*, 2007, pp. 25-25.

[44] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, and X. Zhu, "No power struggles: Coordinated multi-level power management for the data center," in *ACM SIGARCH Computer Architecture News*, 2008, pp. 48-59.

[45] R. Buyya, A. Beloglazov, and J. Abawajy, "Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges," *arXiv preprint arXiv:1006.0308*, 2010.

[46] H. Goudarzi and M. Pedram, "Energy-efficient virtual machine replication and placement in a cloud computing system," in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, 2012, pp. 750-757.

[47] P. Svärd, "Live VM Migration: Principles and Performance". [Online]. Available: http://www.diva-portal.org/smash/get/diva2:707793/FULLTEXT02. [Accessed: 10-Jun-2017]," 2012.

[48] X. Meng, C. Isci, J. Kephart, L. Zhang, E. Bouillet, and D. Pendarakis, "Efficient resource provisioning in compute clouds via vm multiplexing," in *Proceedings of the 7th international conference on Autonomic computing*, 2010, pp. 11-20.

[49] ProLabs, "100G QSFP28 Optical Transceiver". [Online]. Available: http://www.prolabs.com/products/datasheets/msa_standard/QSFP28-100G-SR4-NC.pdf. [Accessed: 04- Jan- 2017]."

[50] Mellanox, "Power Saving Features in Mellanox Products". [Online]. Available: http://www.mellanox.com/pdf/whitepapers/WP_ECONET.pdf. [Accessed: 04- Jan- 2017]."

[51] Enablence, "100GHz WAVELENGTH DIVISION MULTIPLEXER/DEMULTIPLEXER". [Online]. Available: http://www.enablence.com/media/mediamanager/pdf/18-enablence-datasheet-ocsd-awg-standard-100ghzmultidemulti.pdf. Accessed: 04- Jan- 2017]."

[52] Vertatique, "Average Power Use Per Server". [Online]. Available: http://www.vertatique.com/average-power-use-server. [Accessed: 04- Jan- 2017]."

[53] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *ACM SIGARCH Computer Architecture News*, 2007, pp. 13-23.

[54] J. Niemann, "Best practices for designing data centers with the infrastruxure inrow RC," *Application note of American Power Conversion*, 2006.

[55] ARISTA, "ARISTA 7500 Data Center Switch". [Online]. Available: https://www.arista.com/assets/data/pdf/Datasheets/7500_Datasheet.pdf. [Accessed: 04- Jan- 2017]."

[56] GreenTouch, "GreenTouch Final Results from Green Meter Research Study Reducing the Net Energy Consumption in Communications Networks by up to 98% by 2020," *A GreenTouch White Paper*, vol. Version 1, 15 August 2015.

[57] IBM, "Product Guide IBM System x3650 M3". [Online]. Available: https://lenovopress.com/tips0805.pdf. [Accessed: 10-Jun-2017]," 2011.

[58] E. G. Coffman Jr, M. R. Garey, and D. S. Johnson, "Approximation algorithms for bin packing: a survey," in *Approximation algorithms for NP-hard problems*, 1996, pp. 46-93.

[59] K. T. Malladi, B. C. Lee, F. A. Nothaft, C. Kozyrakis, K. Periyathambi, and M. Horowitz, "Towards energy-proportional datacenter memory with mobile DRAM," in *ACM SIGARCH Computer Architecture News*, 2012, pp. 37-48.

[60] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016–2021". [Online]. Available: https://www.google.co.uk/?gfe_rd=cr&ei=gZdeWfuEKcGN8QeJzIHYAg&gws_rd=ssl#q=cisco+vni+2017+pdf&spf=1499371458951. [Accessed 07-Jul-2017]."

[61] Calient. *CALIENT Optical Circuit Switch Brings Sub-60ns Latency to Data Centers ". [Online]. Available: http://www.calient.net/2012/09/calient-optical-circuit-switch-brings-sub-60ns-latency-to-data-center-financial-networks/. [Accessed: 01- Dec- 2016]*.

[62] B. A. Small, A. Shacham, and K. Bergman, "Ultra-low latency optical packet switching node," *IEEE photonics technology letters*, vol. 17, pp. 1564-1566, 2005.

[63] J. Luo, S. D. Lucente, J. Ramirez, H. J. Dorren, and N. Calabretta, "Low latency and large port count optical packet switch with highly distributed control," in *Optical Fiber Communication Conference*, 2012, p. OW3J. 2.

[64] EpiPhotonics, "High-Speed PLZT Optical Switches". [Online]. Available: http://epiphotonics.com/products.html. [Accessed: 04- Jan- 2017]."

[65] B. Meagher, G. Chang, G. Ellinas, Y. Lin, W. Xin, T. Chen, *et al.*, "Design and implementation of ultra-low latency optical label switching for packet-switched WDM networks," *Journal of lightwave technology*, vol. 18, p. 1978, 2000.

[66] Cisco, "Cisco Nexus 3064-X, 3064-T, and 3064-32T Switches".[Online]. Available: http://www.cisco.com/c/en/us/products/collateral/switches/

nexus-3000-series-switches/data_sheet_c78-651097.pdf. [Accessed: 01- Dec- 2016]."

[67] Cisco, "Cisco SFS 7000P Infiniband Server Switch". [Online]. Available: http://www.cisco.com/c/en/us/products/collateral/switches/sfs-7000p-infiniband-server-switch/prod_bulletin0900aecd80337b11.pdf .[Accessed: 01- Dec- 2016]."

[68] M. Technologies, "M4001 16-port 40 and 56Gb/s InfiniBand Blade Switches". [Online]. Available: http://www.mellanox.com/related-docs/oem/dell/Dell_M4001.pdf. [Accessed: 01- Dec- 2016]."

[69] Y. Yang, "Understanding Switch Latency". White Paper, Cisco OSPF,(2009, September). [Online]. Available: http://packetlife.net/captures/protocol/ospf/. [Accessed: 01- Dec- 2016]."

[70] K.-W. Lee, J.-H. Cho, B.-J. Choi, G.-I. Lee, H.-D. Jung, W.-Y. Lee, *et al.*, "A 1.5-V 3.2 Gb/s/pin Graphic DDR4 SDRAM with dual-clock system, four-phase input strobing, and low-jitter fully analog DLL," *IEEE Journal of Solid-State Circuits*, vol. 42, pp. 2369-2377, 2007.

[71] C. Park, H. Chung, Y.-S. Lee, J. Kim, J. Lee, M.-S. Chae, *et al.*, "A 512-mb DDR3 SDRAM prototype with C IO minimization and self-calibration techniques," *IEEE journal of solid-state circuits*, vol. 41, pp. 831-838, 2006.