When Task Scheduling Meets Flexible-bandwidth Optical Interconnects: A Cross-layer Resource Orchestration Design

Xiaoliang Chen[1], Che-Yu Liu[2], Roberto Proietti[3], Shaoyi Chen[1], Zhaohui Li[1], and S.J. Ben Yoo[3]

1. Guangdong Provincial Key Labratory of Optoelectronic Information Processing Chips and Systems, Sun Yat-sen University, Guangzhou 510275, China

2. Department of Computer Science, University of California, Davis, Davis, CA 95616, USA

3. Department of Electrical and Computer Engineering, University of California, Davis, Davis, CA 95616, USA

xlichen@ieee.org, lzhh88@mail.sysu.edu.cn, sbyoo@ucdavis.edu

Abstract: We propose a cross-layer resource orchestration design for task scheduling in flexible-bandwidth optical data center networks. Results show the proposed design can achieve $\sim 8.2\times$, $\sim 1.9\times$ and $\sim 4.8\times$ reductions of request blocking probability, end-to-end delay and packet loss rate, compared with the baseline.

1. Introduction

The explosive growth of cloud applications are pushing today's data centers to evolve to the beyond-Exascale Era. Traditional multi-hierarchy tree-based interconnect architectures employing electronic packet switching [1] are facing enormous challenges in continuing to scale up while sustaining desirable costs (capital and operational) and performance (e.g., delay). In this context, recent studies have reported various low-diameter topology designs (e.g., Xpander [2]), which, when combined with the recent advances in silicon-photonic (SiPh) technologies, enable to build highly energy-efficient and agile (flexible-bandwidth) optical interconnect architectures for data centers [3]. The benefits of such architectures have been explored by previous works, where spatially and temporally nonuniform application traffic was considered [4]. Nevertheless, existing works assume that the traffic distribution data are actively monitored by the control plane or reported by the application layer and ignore the provisioning of the applications (i.e., task scheduling) that essentially determines the traffic profiles.

In this work, we investigate the problem of task scheduling in flexible-bandwidth optical data center networks and propose a cross-layer resource orchestration design. The proposed design adopts a computing&bandwidth resource-aware joint optimization policy for provisioning of virtual machine (VM) requests and performs topology reconfiguration with minimized service disruption when bandwidth demands cannot be meet. Simulation results demonstrate remarkable improvement in task request blocking probability, end-to-end delay and packet loss rate from the proposed design over the baseline.

2. Network Model

We consider a flexible-bandwidth optical data center network of a Hyper-X-like interconnect architecture [4], where servers are grouped into a set of portable data centers (PODs). Fig. 1 shows the layout of a POD of N racks. Each rack consists of M servers assembled with certain amounts of computing/IT resources (i.e., CPU cores, memory, and hard disk). We model the connectivity graph of the POD as $G(V,E)$, where V and E signify the sets of top-of-rack switches (ToRs) and links, respectively. In particular, the ToRs connect to a SiPh switch (e.g., Flex-LIONS [3]) and adopt wavelength division multiplexing (WDM) to realize all-to-all interconnect between racks within the POD. The ToR switches also connect to several other SiPh switches, forming similar fabrics for inter-POD communications. By reconfiguration of the SiPh switches, bandwidth (or wavelength links) can be steered between ToR pairs to offer diversified interconnectivity. In the control plane, the POD controller interfaces with the IT resource manager for cross-layer resource orchestration. We consider a hose task request model [5] (see Fig. 1) denoted by $R(\{c_k, b_k\}|_{k\in[1,K]})$, which features K VM requests, each specifying particular computing ($c_k = [c_k^{cpu}, c_k^{mem}, c_k^{disk}]$) and bandwidth ($b_k$) requirements. Upon the arrival of a request R, the IT resource manager exchanges with the POD controller network configuration and resource utilization states and computes a provisioning scheme by jointly optimizing VM and bandwidth allocation. Let $\{x_{n,k}\}|_{n\in[1,N],k\in[1,K]}$ denote a VM allocation scheme, with $x_{n,k} = 1$ representing VM k being created in rack n (0, otherwise). Then, the bandwidth

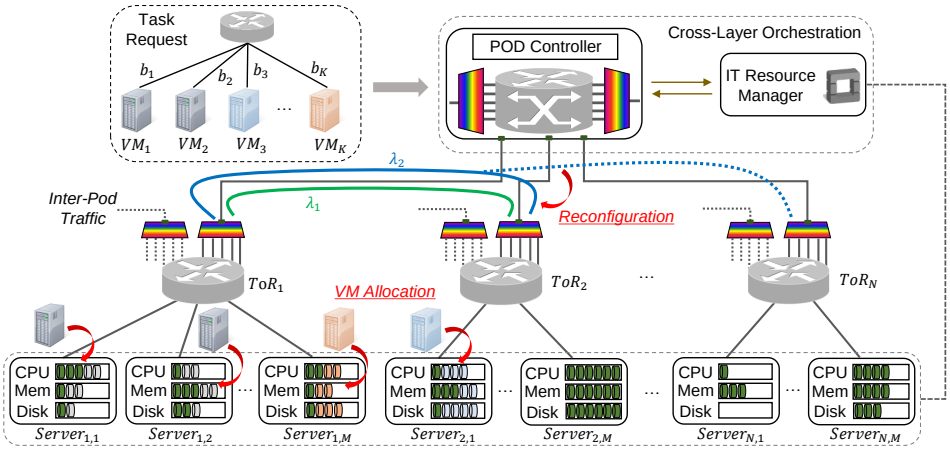


Fig. 1: Principle of task scheduling in a SiPh switch-enabled flexible-bandwidth optical data center network.

demand between each ToR pair can be calculated as,

$$\phi_{n_1,n_2} = \min\left\{\sum_{k\in[1,K]} x_{n_1,k}b_k, \sum_{k\in[1,K]} x_{n_2,k}b_k\right\}, \forall n_1,n_2 \in [1,N]. \quad (1)$$

For the example in Fig. 1, the requested VMs are created with the computing resources in racks 1 and 2, while the POD controller assigns wavelength λ_2, which was originally used to connect ToRs 1 and N, for meeting the increased traffic demand between ToRs 1 and 2.

3. Cross-layer Design

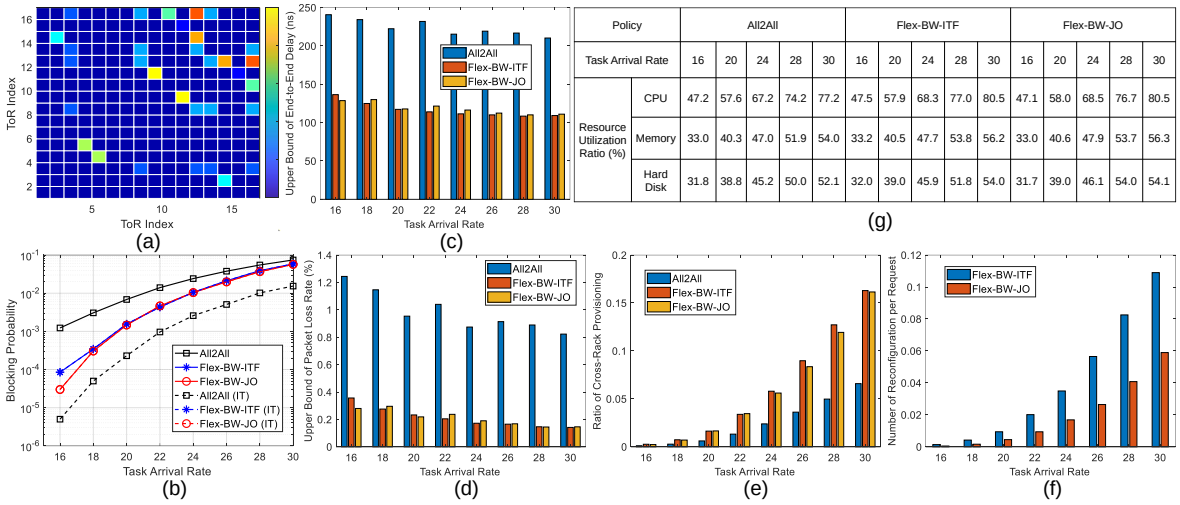
The cross-layer resource provisioning design is composed of two optimization stages.

VM Provisioning. Eq. 1 suggests that allocating VMs across multiple racks to a task incurs inter-ToR communication demands. Therefore, provisioning within a single rack is advocated. A straightforward VM allocation policy is to prioritize selections of racks with more idle computing resources [namely, ITF (IT first)], whose procedures are as follows: *(i)* sort the requested VMs in the descending order of c_k; *(ii)* normalize the amounts of unallocated computing resources in each rack (denoted as C_n) by $C_n/\sum_k c_k$; *(iii)* sort racks in the descending order of $C_n/\sum_k c_k$; *(iv)* for each rack, exhaust the computing resources to provision the pending VM requests before proceeding to the next rack. Note that, ITF fails to incorporate the bandwidth utilization states and thereby can lead to unnecessary reconfiguration operations (thus, increased operational overheads) or even request blocking. To mitigate this issue, we further devise a computing&bandwidth-aware joint optimization policy (namely, JO). JO starts with procedures same as those in ITF. When R cannot be serviced with a single rack n, we estimate the VM allocation in each candidate rack $j \in [1,N]\setminus n$ and obtain a set of intermediate allocation schemes $\hat{x}^j$. The corresponding bandwidth demand $\hat{D}_{j,n}$ can be derived meanwhile. JO assigns each rack j a weight $w_j = (C_j/\sum_{k\in\zeta} c_k)(B_{j,n}/\hat{D}_{j,n}+\theta)$, where $B_{j,n}$ represents the amount of available bandwidth capacity between ToRs j and n, ζ is the set of VMs to be provisioned, and θ is a parameter adjusting the weighting of bandwidth capacity. Afterward, rack j^* with the largest weight is picked. We repeated such operations until ζ is empty.

Bandwidth Reconfiguration. Having obtained the VM allocation scheme $x_{n,k}$, we construct a demand matrix D (if needed) using Eq. 1 and allocate required bandwidth in G. Here, we apply the balanced-load routing scheme discussed in [4]. Alternatively, one can apply any other routing scheme, such as, k-shortest path routing or equal-cost multipath routing. If D cannot be satisfied with G, bandwidth reconfiguration is triggered. With reconfiguration, we aim at accommodating D while minimizing the influences imposed on the existing services. Therefore, *(i)* we first remove links not carrying any traffic and get T_n ports freed in each ToR n. *(ii)* Then, we sort D in the descending order and try to provision each of them on a direct link. Specifically, we can calculate the number of additional wavelengths needed for a ToR pair $<n_1,n_2>$ as $l = \max\{(D_{n_1,n_2} - B_{n_1,n_2})/B_0, 0\}$. The link capacity is augmented by l wavelengths directly if both $T_{n_1}, T_{n_2} >= l$. Otherwise, we seek to reconfigure another $l - \min\{T_{n_1}, T_{n_2}\}$ wavelengths where existing demands are allocated, while ensuring that the total demand affected is less than D_{n_1,n_2} (to secure a minimum gain from reconfiguration). *(iii)* In the case that no feasible direct-link solution exists, we take into account all possible configuration options with T and attempt to route D_{n_1,n_2} on a multi-hop path. *(iv)* Finally, the rest of the idle wavelengths are steered to where bandwidth is highly utilized.

4. Performance Evaluation

We conducted simulations of task scheduling in a POD of 16 ToRs to assess the performance of the proposed design. Each rack is composed of 32 servers. We consulted the server configurations discussed in [6] and set



Policy		All2All					Flex-BW-ITF					Flex-BW-JO				
Task Arrival Rate		16	20	24	28	30	16	20	24	28	30	16	20	24	28	30
Resource Utilization Ratio (%)	CPU	47.2	57.6	67.2	74.2	77.2	47.5	57.9	68.3	77.0	80.5	47.1	58.0	68.5	76.7	80.5
	Memory	33.0	40.3	47.0	51.9	54.0	33.2	40.5	47.7	53.8	56.2	33.0	40.6	47.9	53.7	56.3
	Hard Disk	31.8	38.8	45.2	50.0	52.1	32.0	39.0	45.9	51.8	54.0	31.7	39.0	46.1	54.0	54.1

(g)

Fig. 2: Simulation results: (a) snapshot of demand heatmap; (b) request blocking probability; (c) end-to-end delay; (d) packet loss rate; (e) ratio of tasks serviced with multiple racks; (f) reconfiguration frequency; (g) resource utilization ratio.

each server to be equipped with 32 CPU cores, 256 GB memory, and 3584 GB hard disk. Each task requests $10 \sim 60$ VMs with a random combination of four types, i.e., $[4, 15, 80]$, $[16, 32, 320]$, $[16, 122, 320]$, $[16, 122, 3200]$. The bandwidth demand by each VM is evenly distributed within $[1024, 4096]$ Mbps, while the capacity of each wavelength link was assumed to be 40 Gbps. We compared the cross-layer approaches [with ITF (Flex-BW-ITF) and with JO (Flex-BW-JO)] with a baseline that adopts JO under fixed all-to-all interconnects (denoted as All2All). Fig. 2(a) shows the snapshot of demand heatmap taken from simulations. We can see that the distribution of demand is highly skewed, which confirms the necessity of bandwidth reconfigurability. Figs. 2(b)-(d) show the comparison between the different approaches, where each value was obtained by simulating $200,000$ task requests. First, it can be seen that bandwidth reconfiguration significantly reduces the request blocking probability. In particular, Flex-BW-JO achieves in average $8.2\times$ (up to $41\times$ under task arrival rate of 16) blocking reduction compared with All2All. The benefit from reconfiguration can be revealed by counting the blocking probability due to computing/IT resource scarcity (shown by the dashed curves in Fig. 2(b)). We can observe that almost all request blocking in Flex-BW-ITF and Flex-BW-JO is due to lacking of computing resources, whereas this ratio for All2All is less than 10% in average, indicating that bandwidth reconfiguration can effectively alleviate link bottlenecking in data centers. Figs. 2(c)-(d) show the results of end-to-end packet delay and packet loss rate obtained by considering extreme cases where VMs send data in full rates and by applying a queuing model presented in [7] (packet size set as 296 Byte and buffer size set as 15 packets). On average, compared with All2All, Flex-BW-ITF and Flex-BW-JO reduce the end-to-end delay and packet loss rate by $\sim 1.9\times$ and $\sim 4.8\times$, respectively. Fig. 2(e) shows the ratios of tasks serviced with multiple racks. As expected, bandwidth reconfiguration enables more tasks being provisioned across racks. Fig. 2(f) shows the results of reconfiguration frequency from Flex-BW-ITF and Flex-BW-JO. Despite that the two approaches achieve similar performance in request blocking probability, delay, and packet loss rate, Flex-BW-JO requires $\sim 2.2\times$ less reconfiguration operations than Flex-BW-ITF. This is because Flex-BW-JO selects racks with more interconnect bandwidth during VM provisioning so that the bandwidth demands are more likely to be meet by the existing configurations. Finally, we summarize the utilization ratios of different types of computing resources in Fig. 2(g).

5. Summary

This paper proposes a cross-layer resource provisioning design for task scheduling in flexible-bandwidth optical data center networks. Simulation results verify the benefit of the proposed design.

References

1. "Top500", https://www.top500.org/.
2. A. Valadarsky *et al.*, *Proc. of CoNEXT*, 205–219 (2016).
3. X. Xiao *et al.*, *J. Sel. Topics Quantum Electron.*, **26**, 1-10 (2020).
4. X. Chen *et al.*, *J. Opt. Commun. Netw.*, **13**, C10-C20 (2021).
5. H. Ballani *et al.*, *ACM SIGCOMM Comput. Commun. Rev.*, **41**, 242-253 (2011).
6. X. Dai *et al.*, *Trans. Cloud Comput.*, **4**, 210-221 (2016).
7. K. Kosek-Szott, *Proc. of IEEE PIMRC*, 74-79 (2003).

This work was supported in part by NSFC Project U2001601, NSF ECCS Award #1611560, and NSFC Project 62035018.