Evaluation Study of a Proposed Hadoop for Data Center Networks Incorporating Optical Circuit Switches

Xiaoyu Wang, Malathi Veeraraghavan, and Haiying Shen

***Abstract*—Hybrid data center network architectures, which combine electrical packet switches (EPSs) with optical circuit switches (OCSs), have been proposed to handle the inter-rack link oversubscription problem in a cost- and power-efficient manner. However, the high reconfiguration delay of optical switching technologies makes it challenging to effectively leverage OCSs in these hybrid networks. In our prior work, we proposed a modified version of Hadoop, called Hadoop for hybrid networks (HHN), to create network traffic patterns that match the characteristics of OCSs. This paper presents a comprehensive comparative evaluation of HHN on hybrid networks and the original Hadoop on conventional EPS-only networks. Numerical results validate our hypothesis that it is feasible to achieve similar system-level and user-level performance with HHN while simultaneously achieving power and cost savings with the hybrid network. When the percentage of shuffle-heavy (SH) jobs is small, e.g., 5%, HHN performance is the same as that of the original Hadoop on an EPS-only network. When the percentage of SH jobs is large, e.g., 20%, the HHN performance is almost the same even at high loads, and even with a smaller number of input-block replicas when we placed an upper bound on the per-job input-data size. For large SH jobs with input-data sizes larger than the upper bound, the performance of HHN could be improved significantly when storing a larger number of replicas for each input block.**

***Index Terms*—Hadoop; Hybrid data center networks; Optical circuit switching.**

I. Introduction

Conventional fat-tree-based data center networks (DCNs), consisting of only electrical packet switches (EPSs), are commonly oversubscribed at the higher layers. While there is full bisection bandwidth within a rack, inter-rack bandwidth is typically a fraction of intra-rack bandwidth. This design choice is made to limit capital expenditures (CAPEX) and operating expenditures (OPEX).

Big-data applications are constrained by such oversubscribed networks when they involve network-intensive operations such as shuffle or join in which large amounts of data are transferred between racks. Consider, for example, Hadoop [1], which is an implementation of the MapReduce framework and used extensively for analysis of large data sets. Map tasks are typically run on hosts where input data blocks are stored. As the input data are spread over the cluster randomly in the distributed Hadoop file system, a subsequent shuffle phase is required to move map-task output to hosts on which reduce tasks are scheduled. This shuffle phase often requires data movement across oversubscribed inter-rack links. To overcome this problem, research papers [2–4] proposed methods to reduce inter-rack traffic by having the resource scheduler optimize task-placement strategies. For example, ShuffleWatcher [3] attempts to localize shuffling to one or a few racks but sometimes requires inter-rack communications to move input data.

An alternative approach to solving this problem is to add optical circuit switches (OCSs) to create hybrid EPS-OCS DCNs. Proposals for such hybrid DCN architectures [5–9] have been evaluated and shown to offer higher inter-rack capacity in a cost- and power-efficient manner when compared with EPS-only DCN designs.

While OCSs offer cost and power advantages over their EPS counterparts, OCSs suffer from a disadvantage of requiring high reconfiguration delays (e.g., μs-to-ms [8]). Such delays are significantly higher than the budget allowed for packet switching. The implication of this disadvantage is that dynamic optical circuits can only be used for data transfers that are large enough to make the circuit setup delay a small fraction of the data transmission (emission) delays.

The above-cited prior work on hybrid networks [5–9] uses techniques such as buffering packets at top-of-rack (ToR) switches to collect a sufficiently large amount of data before dynamically provisioning an optical circuit between two ToR switches. But without an application-level view of traffic demands and dependencies, circuit utilization and application performance could be poor [10].

Therefore, in recent work [11], we proposed a set of adaptations for Hadoop named Hadoop for hybrid networks (HHN). HHN is a "cross-layer" approach that modifies the application to match its network traffic better to hybrid EPS-OCS networks and dynamically configures optical circuits for the application when needed. The required

Manuscript received February 21, 2018; revised June 24, 2018; accepted June 26, 2018; published July 19, 2018 (Doc. ID 323410).

X. Wang (e-mail: xw5ce@virginia.edu) and M. Veeraraghavan are with the Department of Electrical and Computer Engineering, University of Virginia, Charlottesville, Virginia 22904-4743, USA.

H. Shen is with the Department of Computer Science, University of Virginia, Charlottesville, Virginia 22904-4743, USA.

https://doi.org/10.1364/JOCN.10.000C50

changes in Hadoop are in input data block placement, map-task scheduling, reduce-task scheduling, and shuffling. The key idea is to concentrate blocks of data sets that are likely to receive shuffle-heavy (SH) Hadoop MapReduce jobs to a few racks and to assign map tasks to containers on these racks so that map output is concentrated on a few racks. Correspondingly, reduce tasks are also assigned to containers in a limited number of racks so that map output can be shuffled from map racks to reduce racks via high-speed dynamic optical circuits.

In this paper, we present a simulation-based evaluation of HHN on hybrid networks and compare its performance with that of the original Hadoop on EPS-only networks. Portions of this work were presented at the 2017 IEEE GLOBECOM conference [12]. For this paper, we extended our prior work as follows: (i) quantified the cost- and power-savings achievable in hybrid networks when compared with EPS-only networks; (ii) designed and validated an approach for improving the performance of very large SH jobs in HHN; (iii) tested our performance-comparison conclusions by running simulations with multiple traces; and (iv) demonstrated the benefits of shuffle-phase decoupling in our proposed HHN.

Our simulation approach for performance comparison of HHN and the original Hadoop on EPS-only networks consisted of (i) creating a variety of workloads by mixing synthetic regular jobs (those with a shuffle size less than 2 GB) with SH jobs drawn from the real-world Facebook-2010 traces [13], with different ratios of regular jobs to SH jobs; (ii) using different system (network) configurations, e.g., four-rack and 12-rack systems, and 75 and 100 for the percentage of ToR switches that are connected to the OCS in the hybrid network; and (iii) changing the job arrival rate to study the system under high levels of CPU utilization. The performance metrics used for the comparative evaluation included system metrics, such as makespan and CPU utilization, and per-job metrics, such as response times and unfairness.

Given the cost and power advantages of hybrid networks over EPS-only networks, we identified workloads in which HHN performance was worse than the original Hadoop performance on EPS-only networks and characterized the performance degradation under different system configurations and different CPU loads. We then evaluated the benefit of changing a critical Hadoop parameter, i.e., the number of replicas used for storing data sets. A small increase from two to three yielded significant performance improvements for HHN.

Our key findings are as follows: (i) hybrid networks can achieve significant savings in cost and power consumption when compared with EPS-only networks; (ii) restricting SH data sets to a few racks in order to concentrate map output so that optical circuits can be used in the shuffle phase of MapReduce jobs can cause increased waiting delays for containers and, consequently, increase SH job response times and job unfairness; (iii) as a consequence, if the percentage of SH jobs in a workload is high, e.g., 20%, or there are large SH jobs (i.e., jobs that require processing of large data sets), the limitation on the number of racks from which containers can be assigned to SH-job map tasks could result in longer makespans and lower CPU utilization because hosts in racks that do not have SH data sets could be idle; (iv) the relative degradation of per-job response times in HHN is smaller in larger systems, i.e., networks with more racks, and lowering the percentage of ToR switches connected to the core OCS favors regular jobs over SH jobs and vice versa; (v) HHN performance can be improved significantly by increasing the number of input-block replicas even by just one, e.g., from two to three; and (vi) in small systems at high loads, without container preemption, the original Hadoop could enter a deadlock leading to significantly longer response times and makespan, while, in contrast, HHN handles the problem elegantly by decoupling the shuffle phase from reduce tasks.

The rest of the paper is organized as follows: Section II provides the reader background information on Hadoop and reviews related work. Section III reviews our proposed modifications to Hadoop for matching traffic to the characteristics of hybrid networks. Our comparative evaluation of this modified Hadoop with the original Hadoop is presented in Section IV. Section V presents our conclusions.

II. Background and Related Work

Section II.A offers a brief tutorial on Hadoop. Section II.B reviews other work on how hybrid EPS-OCS DCNs are used and on Hadoop techniques for avoiding inter-rack network traffic in EPS-only DCNs.

A. Background

Hadoop is used for storing and processing large data sets [1]. Hadoop has three elements: (i) Hadoop distributed file system (HDFS) for storage support; (ii) yet another resource negotiator (YARN) for resource scheduling; and (iii) MapReduce for processing stored data sets.

Each MapReduce job has an application master (AM). For a job to access its input data, the AM locates the nodes on which blocks of the data set are stored. The AM then submits requests for containers to YARN. Hadoop uses the approach of "bringing-code-to-data" rather than "data-to-code." Thus, transfers of input data blocks are avoided. However, data movement cannot be avoided in the shuffle phase because map output needs to be moved to the nodes on which reduce tasks are scheduled.

To decrease the impact of communication delay in the shuffle phase, a technique called "reduce slow start" is used in which reduce tasks are initiated after a specified percentage (default: 5%) of map tasks have completed [1]. This ensures that the output of completed map tasks is transferred to the nodes on which reduce tasks are scheduled, even while other map tasks are running. This approach reduces total job execution time by hiding most of the shuffle delay. But a disadvantage is that CPU resources could be wasted if reduce tasks have to wait for map tasks to complete.

B. Related Work

Hybrid electrical/optical data center network architectures using OCS include Helios [5], *c*-Through [6], OSA [7], Mordia [8], and REACToR [9], among others. Free-space optics-based reconfigurable interconnects, e.g., FireFly [14], Diamond [15], Graphite [16], and ProjecToR [17], have also been proposed for DCN. In most of these approaches, traffic is aggregated and/or monitored to determine the pairs of ToR switches that should be interconnected. For example, ProjecToR proposes to derive probabilities that two ToR switches will communicate from historical traffic matrices. While the advantage of these approaches is that applications do not need to be modified, unlike in our approach, the disadvantage is that these solutions cannot fully leverage optical circuits.

Yamashita *et al.* [18] proposed a Hadoop-triggered hybrid data center orchestration architecture for reducing power consumption. The architecture identifies shuffle-heavy jobs by estimating shuffle data sizes and redirects the shuffle traffic onto optical circuits. While this approach uses application-level information to trigger circuit setup, reconfiguring circuits for small shuffle flows may introduce high reconfiguration overhead.

A third track of related work comes from the cloud computing research community. Besides delay scheduling [2], there are other solutions that try to avoid inter-rack transfers. The ShuffleWatcher solution [3] proposes to monitor network traffic and to use shuffle-aware maps and reduce task placement algorithms in a manner that reduces shuffle traffic. Another network-aware scheduling approach [4] proposes handling large jobs with predictable job characteristics with an offline planned scheduling solution to reduce shuffle traffic. Wang *et al.* [19] propose scheduling reduce tasks near the nodes where map output is generated so that inter-rack shuffle traffic can be reduced. Hybrid EPS-OCS networks, with dynamic circuit management, offer an alternative solution to this inter-rack shuffling problem.

III. Our Proposed Hadoop for Hybrid Networks

Figure 1 illustrates an example hybrid network architecture for which HHN is designed. Lower-rate (10 GE) links that connect ToR EPSs and the core EPS are used for general-purpose traffic, while higher-rate (100 GE) links between ToR EPSs and the OCS are used in ToR-to-ToR

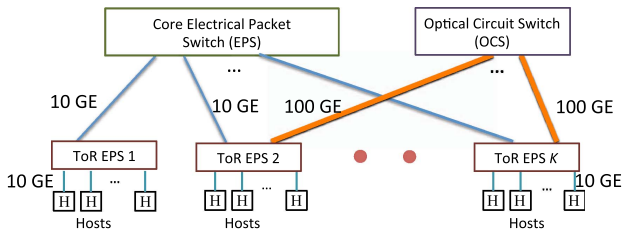

Fig. 1. Example of hybrid EPS-OCS data center network (DCN) architecture.

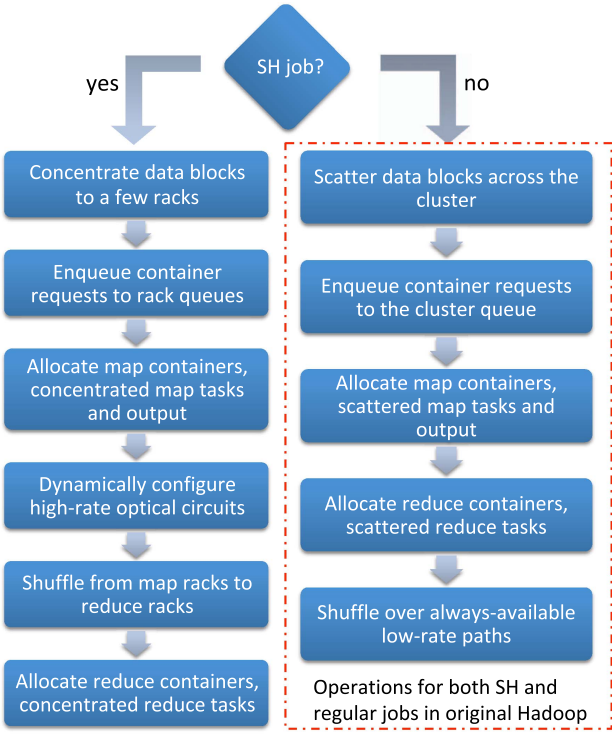

Fig. 2. Flow chart of HHN.

dynamic optical circuits setup/released through the OCS by a controller (not shown in Fig. 1). Not all ToR switches need to be connected to the OCS, e.g., ToR1 is not connected to the OCS. Our model assumes that ToR EPSs in K_o racks in a system of K racks, where $K_0 \leq K$, are connected to the OCS.

To make Hadoop work effectively in such a hybrid network, we proposed modifications to the following [11]: (i) how a data set is stored by HDFS; (ii) how the scheduler assigns containers to map tasks of SH jobs; (iii) how the scheduler assigns containers to reduce tasks of SH jobs; and (iv) how map output is shuffled over optical circuits. The different operations for SH and regular jobs in HHN are summarized in Fig. 2.

The starting point is that, during HDFS storage, blocks of an SH data set, which is a data set that is likely to be processed by SH jobs, are concentrated to a few racks. User-provided input or historical job-data analysis can be used to determine which data sets are likely to receive SH jobs.

Next, map tasks are scheduled only on nodes in racks where the SH data set blocks are stored (this avoids input-data movement between racks just as in the original Hadoop). With data set replication (HDFS default is three replicas), map tasks could be executed on any rack containing the required blocks. Nevertheless, because the number of containers on a rack is limited (when compared with the number of containers across the whole cluster), the wait times for map tasks of an SH job could be high. Therefore, we propose maintaining per-rack queues (see Fig. 3), in addition to the cluster queue, and giving priority

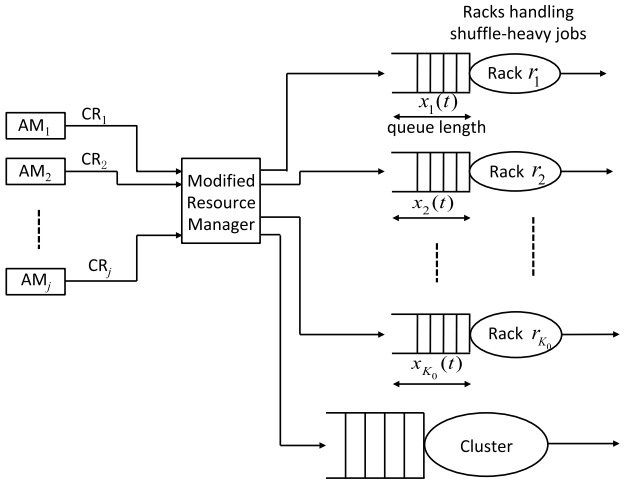

Fig. 3. Per-rack queueing for shuffle-heavy jobs in HHN. AM, application master; CR, container requests for map and reduce tasks.

to SH jobs by submitting task-container requests for these jobs to the corresponding per-rack queues. All the container requests (CR) for regular jobs are submitted to the lower-priority cluster queue. A free container on a rack r is allocated to a regular job only when the per-rack queue of rack r is empty. System parameters (such as the number of racks to which SH datasets are limited) will impact fairness of resource allocation between regular jobs and SH jobs.

By concentrating blocks of an SH data set to a few racks and running map tasks on these racks, map output becomes concentrated on a few racks. The goal is to make the size of map output on a rack large enough so that the transfer delay is several times longer than the optical circuit setup delay. This allows the shuffling to occur over an optical circuit that has been dynamically setup between two ToR switches via the OCS (see Fig. 1).

But before shuffling can occur, YARN must assign containers for the reduce tasks. Just as map tasks were assigned containers on a few racks (i.e., the few racks on which the job's data set blocks were stored), reduce tasks must also be concentrated to a few racks so that rack-to-rack map output can be transferred on an optical circuit. To avoid a co-scheduling problem among map task completions, reduce-task container assignments, and optical-circuit availability, the shuffling phase is decoupled from the reduce tasks. As soon as YARN notifies the job's AM of the racks on which it plans to assign containers for its reduce tasks, the AM initiates the shuffling over an optical circuit using node managers on the map and reduce racks.

When the shuffling is completed, and all the map output is available on the planned reduce-task racks, YARN makes the container assignments for the reduce tasks.

In summary, HDFS, YARN, application managers, and node managers are modified in HHN, so that MapReduce jobs can successfully leverage the high-bandwidth optical circuits of hybrid networks.

IV. Evaluation

We evaluate the performance of the HHN solution by comparing it against the original Hadoop in an EPS-only network. To achieve a fair comparison, we assume that the ToR-to-core links in the EPS-only network have the same capacity as the transceiver rates in the OCS segment of the hybrid network. Our hypothesis is that, compared with the EPS-only network, the HHN solution can offer almost equivalent job performance but with power and cost savings.

To test the hypothesis, we first analyze the price and power consumption of the two types of DCN architectures (see Section IV.A), then conduct a detailed simulation study to compare job performance, which could potentially be worse in the HHN solution. In the EPS-only network, all links are of high-rate and are always available; in contrast, in the hybrid network solution, the high-rate circuits have to be set up dynamically across the OCS when needed; therefore, we expect job performance to be worse in HHN.

The purpose of our simulation is to quantify job performance and recommend parameter settings to achieve the same level of performance as with the original Hadoop on EPS-only networks. Results showed the validity of our hypothesis.

Section IV.A compares the price and power consumption of hybrid and EPS-only DCN architectures. Section IV.B describes our simulator, input parameters, workloads, and evaluation metrics. Section IV.C provides an in-depth analysis of a single SH job execution, while the remaining subsections present simulation studies with multiple SH and regular jobs. Section IV.D presents the results of our comparison of HHN and the original Hadoop for a baseline setting of system parameters. Section IV.E presents the effects of changing two key system parameters. Section IV.F presents the effect of changing one key Hadoop parameter. Section IV.G presents generalized results for multiple traces with the same parameter settings.

A. Power and Cost Evaluation

This subsection presents a differential power and cost comparison of example hybrid and EPS-only DCNs. Because the downlink ToR switch ports (ports connected to the servers) are the same in all DCNs, these ports are omitted from the comparison.

Two configurations of the hybrid architecture with K ToR switches, as illustrated in Fig. 1, are modeled here: hybrid 100% and hybrid 75%, where the 100% and 75% values denote the percentage of ToR EPS connected to OCS in the two configurations, respectively.

Table I lists the number of different types of ports in the hybrid 100%, hybrid 75%, and EPS-only DCNs. The OCS is present only in the hybrid DCNs, and the number of OCS ports in these hybrid DCNs depends on the percentage of ToR EPS connected to the OCS. The total number of 10G EPS ports (including all the ToR switch and core EPS

TABLE I
INPUT PARAMETERS FOR A COMPARISON OF THREE DCNS

Components	Number of Ports Needed in Different Architectures			Price (USD) per Port	Power (W) per Port
	Hybrid 100%	Hybrid 75%	EPS-only		
OCS port[a]	K	$0.75K$	—	400	0.15 (lower), 0.6 (higher)
10G EPS port	$2K$	$2K$	$2K$	—	—
10G SR transceiver	$2K$	$2K$	$2K$	—	—
100G EPS port[b]	K	$0.75K$	$2K$	900 (lower), 2000 (higher)	12.5 (lower), 42 (higher)
100G SR transceiver[c]	K	$0.75K$	$2K$	2500	1.5
Fiber[d]	$2K$	$1.75K$	$2K$	13	0

[a]The price and power consumption values for an OCS port were obtained from Calient for the S320 OCS [20] and for Glimmerglass Intelligent Optical System 600 [21], respectively.
[b]The lower and higher prices for a 100G EPS port were obtained for Arista 7160 and 7280SRAM-48C6 [22], respectively. Power consumption values were obtained from data sheets for Cisco Nexus 7700 [23], Juniper QDX10002 [24], Arista 7280SRAM-48C6 [22], and Huawei CloudEngine 12800 [25].
[c]The price and power consumption values for a 100G transceiver were obtained for Arista QSFP-100GBASE-SR4 [26] and Cisco QSFP-100G-SR4 [27], respectively.
[d]The price of fiber was obtained from fs.com [28].

ports) is $2K$ in all three DCNs. For a fair comparison, we assumed that the ToR-to-core capacity in the EPS-only DCN is 110G (100G + 10G) per ToR switch to match the total ToR-to-core capacity in the hybrid 100% architecture. The number of 10G and 100G transceivers is the same as the number of 10G and 100G ports, respectively. The number of 100G EPS ports is $2K$ in the EPS-only DCN because 100G ports are required in the ToR switches (for the uplinks) and the core EPS. But, in the hybrid DCNs, the OCS ports terminate the 100G uplinks from the ToR switches; hence, only K and $0.75K$ 100G EPS ports are required at the ToR switches, in the hybrid 100% and hybrid 75% DCNs, respectively. The number of fiber links required in the EPS-only DCN is $2K$ because, as stated above, we assumed that each ToR switch has two uplinks: 10G and 100G. In the hybrid DCNs, one fiber is required from each ToR switch to the core EPS, and a second fiber is required from each of the ToR switches that is connected to the OCS.

Next, we explain how we obtained the price and power-consumption values listed in Table I. The OCS-port price was obtained in June 2018 for a 320×320 switch. Because the total number of 10G EPS ports is $2K$ in all three DCNs, the price and power-consumption values are not required for our differential comparison and thus not listed in Table I. Because price and power consumption values are variable, we offer two values and mark them as "lower" and "higher." These values are not necessarily the minimum and maximum values because some vendors offer discounts, and other vendors offer products without warranties or maintenance contracts. Table I shows two June 2018 prices for a 100G EPS port: $900 and $2000, which correspond to per-port prices of a standard switch versus a deep-buffer switch. The amount of buffer space and the switch sizes account for the difference in per-port prices. Transceiver prices vary significantly. Third-party vendors offer lower-priced transceivers but without warranties. The transceiver price listed in Table I was obtained directly from a switch vendor in June 2018. All prices are retail values, including warranties, and are without discounts.

Table II compares the cost and power consumption of the hybrid and EPS-only DCNs, assuming a system size of 100 racks. We present a baseline cost and power-consumption value for the EPS-only DCN (these values do not represent total price or total power consumption because 10G ports were omitted) and the savings achieved in the hybrid DCNs when compared with the EPS-only DCN. The cost savings of the hybrid 100% DCN over EPS-only network were $300,000 and $410,000, when using the lower and higher values for component prices, respectively. For the hybrid 75% DCN, the cost savings are even higher. Similarly, the power savings of hybrid 100% and hybrid 75% DCN over EPS-only DCN are 4.3 and 5.4 kW, respectively, when using the higher numbers for component power-consumption values. The additional cost and power savings of hybrid 75% DCN over the hybrid 100% DCN come from the smaller number of OCS ports needed in the hybrid 75% DCN. Finally, because optical switches generate less heat than electrical switches, hybrid DCN architectures can achieve additional cost savings in cooling systems.

B. Simulation Methodology

Simulator. We implemented an event-based simulator model of HHN. The HDFS-simulation module allows all blocks of an SH data set to be stored in a specified number of racks. The YARN-simulation module supports per-rack queues for SH jobs to request containers for map and reduce tasks, while regular jobs enqueue their container

TABLE II
COST AND POWER CONSUMPTION COMPARISON OF THREE 100-RACK DCNS

Architectures	Cost (USD)		Power (kW)	
	Lower	Higher	Lower	Higher
EPS-only (baseline)	$\$682.6K$	$\$902.6K$	2.8	8.7
Hybrid 100% (savings)	$\$300.0K$	$\$410.0K$	1.4	4.3
Hybrid 75% (savings)	$\$470.3K$	$\$532.8K$	1.7	5.4

requests in a cluster queue. Hadoop fair scheduler with delay scheduling is used to allocate containers for tasks in the cluster queue.

The network model is fairly coarse. The required optical links are assumed to be available whenever needed. To estimate the time to move map output for SH jobs, the size of map output is divided by the optical network transceiver rates, and a switch reconfiguration delay of 10 ms is added. For map output of regular jobs, which is transferred on paths traversing only the ToR and core EPS, flow instantaneous rates are computed by dividing the rate of the link carrying the maximum number of concurrent flows by the number of flows. Flow rates are updated every 1 ms. Map output transfer time for regular jobs is computed from the total map output size and the per-ms transfer sizes.

For comparison, we also simulated the original Hadoop on an EPS-only DCN. In the original Hadoop system, resource requests of both shuffle-heavy jobs and regular jobs are enqueued in a single cluster queue, which is served by the Hadoop fair scheduler with delay scheduling.

The simulator, written in Python, has 1000+ lines of code. All simulation runs were executed on a University of Virginia HPC cluster called Rivanna [29].

Parameters. The default values of simulation parameters are shown in Table III. Unless otherwise specified, these default values are used in all the runs. Specifically, we chose the EPS-only link rates (110 Gbps) to be the sum of the EPS-link (10 Gbps) and OCS-link rates (100 Gbps) of the hybrid network. The intra-rack link rate used in computing transfer times is 8 Gbps because we assumed the background traffic rate to be 2 Gbps on the 10 Gbps intra-rack links.

Workloads. We started with the Facebook 2010 (FB-2010) workload, which provides the following information for each job: (i) arrival time instant; (ii) input data set size; (iii) shuffle data size; and (iv) reduce output size. Assuming that each map task processes one input block of size 128 MB, and that the number of reduce tasks is equal to the number of map tasks divided by 8, we derived the number of map tasks and number of reduce tasks for each job from the size of its input data set.

TABLE III
SIMULATION PARAMETERS

System Parameters	Value
Number of racks	2[a], 4, 12
Number of hosts per rack	20
Number of containers per host	16
K_o/K (percentage of ToR EPS connected to OCS)	75%, 100%
Intra-rack link rate	8 Gbps
Inter-rack EPS link rate in hybrid network	10 Gbps
Optical link rate in hybrid network	100 Gbps
Inter-rack link rate in EPS-only network	110 Gbps
Hadoop parameters	**Value**
Number of replicas of each input block	2, 3[b]
Reduce slow start	90%

[a]Only in Section IV.C.
[b]Number of replicas set to three only in Section IV.F.

TABLE IV
TRACE COMPOSITION; RJS, REGULAR JOB SETS

	Percentage of Job Types	
Number of Maps	RJS1	RJS2
1–9	40%	20%
10–99	40%	50%
100–499	18%	28%
500–10,000	2%	2%
Shuffle size		
0		10%
0–0.8 GB		70%
0–2 GB		20%
	TS1	TS2
Regular jobs	RJS1	RJS2
SH jobs in Sections IV.D–IV.F	First 40 SH jobs in FB-2010 workload	First 60 SH jobs in FB-2010 workload with input size <800 GB
SH jobs in Section IV.G	—	Randomly picked SH jobs in FB-2010 workload with input size <800 GB

In the FB-2010 workload, more than 50% of the jobs are small jobs, with only one or two map tasks. With the given job arrival times, the original workload results in low CPU utilization, i.e., around 10%, in our simulation. To achieve higher CPU utilization levels, we generated two trace sets, TS1 and TS2, which consist of larger (artificially created) regular jobs and SH jobs that were directly drawn from the FB-2010 workload. The composition of two regular job sets (RJS) is shown in Table IV. Uniform distributions are used to select the group (based on number of map tasks) and the specific number of map tasks within the selected group. Uniform distribution is used similarly to select the shuffle size of a regular job. We defined jobs with a shuffle-data size larger than 2 GB as shuffle-heavy jobs because the duration to transfer this data on 100 Gbps links is sufficiently longer than the 10 optical circuit setup delay overhead. The compositions of trace sets TS1 and TS2 are shown in Table IV. We used the first 40 SH jobs from the FB-2010 workload in TS1. For TS2, we included the first 60 SH jobs whose input-data sizes were smaller than 800 GB because we found that one very large SH job can skew the results, as described in Section IV.D.

Each trace in the trace sets was generated by varying two parameters, i.e., job arrival rate λ and SH-job percentage p_s. The inter-arrival times of jobs were decided by drawing samples of an exponentially distributed random variable with parameter λ. The SH-job percentage p_s was used to set the percentage of SH jobs in a trace. For each job, a Bernoulli distributed sample with parameter p_s was drawn to decide whether the job should be an SH job or a regular job. If it was an SH job, then its parameters were taken from one SH job in the FB-2010 workload following certain rules, as shown in Table IV. For example, all traces in TS1 have the same 40 SHJs, in the same relative order. When generating traces for TS2, the SH jobs in each trace are randomly selected from the SH jobs in the FB-2010 workload that have input-data sizes smaller that 800 GB.

Evaluation metrics. We used two types of metrics, i.e., per-job metrics and system metrics. Per-job metrics include job response time and per-job unfairness. The system metrics used to characterize the overall performance of the system are makespan and CPU utilization.

Job response time is defined as $t_j^c - t_j^a$, where t_j^a is the arrival instant of job j and t_j^c is the job completion time. Per-job unfairness is defined as:

$$f_j = \int_{t_j^a}^{t_j^c} \max\left\{d_j(t) - \frac{a_j(t)}{R}, 0\right\} \mathrm{d}t, \tag{1}$$

where:

$$d_j(t) = \min\left\{\frac{1}{N(t)}, \frac{r_j(t)}{R}\right\}. \tag{2}$$

At time instant t, the percentage of resources deserved, $d_j(t)$, by job j (from a fairness point of view), depends upon the number of jobs $N(t)$ in the system and the ratio of the resources requested $r_j(t)$ to the number of system resources R. For example, if a system has only two jobs, and one job requires five containers, and the second job requests and receives the remaining 10 containers in the system, the percentage of resources deserved by the first job is 1/3, not 1/2. The instantaneous unfairness of a job is the difference between the amount of its deserved resources and the amount of its allocated resources, $a_j(t)$. Per-job unfairness f_j is computed by integrating its instantaneous unfairness over the job's lifetime.

Makespan is defined for a trace consisting of J jobs as $t_J^c - t_1^a$. CPU utilization for a trace of J jobs is the average utilization of all containers in the system over the time period of $[t_1^a, t_J^c]$.

C. Effect of Clumping on a Single Shuffle-Heavy Job

Using the modified data-block placement policy, the input data set of a shuffle-heavy job is concentrated to a few racks, which limits the amount of computing resources accessible for the job. To study the effect of input-data clumping, we start with a single shuffle-heavy job in a small system. We simulated a cluster of two racks, in which there are a total of 16 containers indexed from 0 to 15. The SH job consists of 36 map tasks and five reduce tasks. In HHN, the input data set is stored only in the first rack, while the data set is stored in both racks for the original Hadoop. The optical link rate is 5 Gbps in HHN. The inter-rack electrical link rate is 500 Mbps in the hybrid network and 5.5 Gbps in the EPS-only network. Here, we use lower link rates when compared with the values listed in Table III because we simulate only one job and use a smaller system (with only 16 containers).

Figure 4 illustrates how containers are allocated to the SH job when it runs on the two networks. The dashed line represents the AM container. The thin and thick lines in green correspond to map-task containers and reduce-task containers, respectively. The black segments show the time period when map output is being shuffled. The job is completed faster in the original Hadoop than in HHN (75s versus 113s). This is because the job can only use the eight containers in the first rack to execute map tasks due to its concentrated input data set in HHN, while it can use all 15 containers (except for container 0 used by the AM) to execute map tasks. On the other hand, thanks to the decoupled shuffle phase from reduce tasks, reduce containers do not need to sit idle when waiting for the shuffle phase to finish with the modified Hadoop [see shorter black segments in Fig. 4(a) than in Fig. 4(b)]. Next, we study how multiple SH jobs and regular jobs interact.

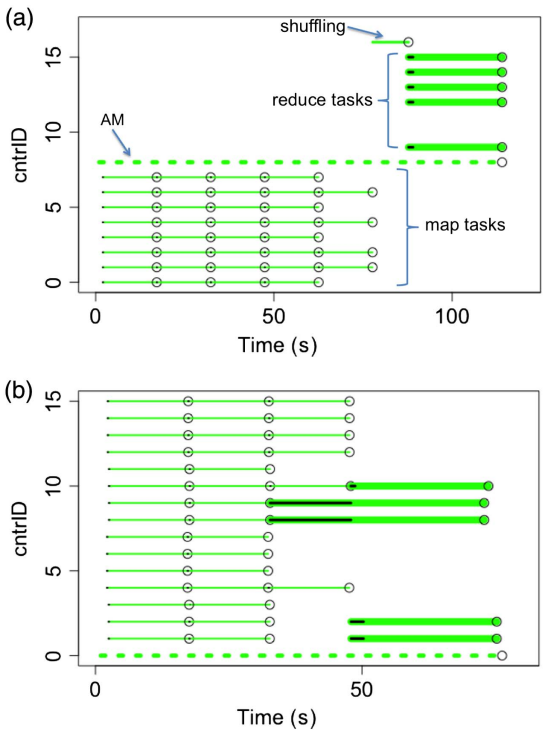

Fig. 4. Start and finish times of map tasks, reduce tasks, and shuffling of a single shuffle-heavy job. (a) HHN in the hybrid network. (b) Original Hadoop in the EPS-only network.

D. Comparison in a Baseline Setting

The system parameters and Hadoop parameters in this baseline setting are as specified in Table III, with the number of racks set to 12 and the percentage of ToR EPS connected to OCS set to 75%. The notation "HHN 75%" is used to represent this configuration.

The job traces used for these runs were generated with TS1 settings. A total of nine different traces were generated by combining three values of λ and three values of p_s. Generation of jobs for a trace was terminated when the 40 SH jobs from the FB-2010 workload were included as per our TS1-workload specification (see "Workloads" paragraph in Section IV.B).

Table V first shows two trace properties and then compares makespan and CPU utilization for the two Hadoop versions. The last-job arrival time is useful to interpret the makespan. The reason for the difference in the number of jobs in traces is as follows. When SH jobs constitute only 5% of the trace, approximately 800 jobs were required before the 40 SH jobs from FB-2010 could be included, while, with 10% and 20% of the trace being SH jobs, approximately 400 and 200 jobs, respectively, were required to include the 40 SH jobs.

System metrics. The makespan in the two solutions, HHN 75% and original Hadoop (in the EPS-only network), are almost the same when the trace has a large number of jobs, e.g., 800. This is because, in all nine traces (recall that the same 40 SH jobs were included in all TS1 traces), there was one large SH job with 19,000 map tasks, which arrives in the second half of the traces. In HHN, each SH job can use containers in only two racks (because each data set has only two replicas; see Table III). This results in longer job response times for SH jobs than in the original Hadoop solution where SH jobs can use containers in any rack.

With the longer traces, this large SH-job response time was hidden by the large number of jobs that came after it, resulting in the same makespan. However, with shorter traces (i.e., the 200-job traces), the large SH job was still running when all the other jobs had completed. The makespan difference between the two networks is 324 s under the setting $p_s = 20\%$ and $\lambda = 0.6$/s. This difference is almost equal to the job response time difference for the large SH job.

Table V also shows CPU utilization of the two types of systems for each trace. When the percentage of SH jobs in the trace is small, e.g., 5%, CPU utilization is the same in the HHN 75% and original Hadoop solutions because regular jobs dominate, and these jobs can be assigned containers in any rack. But when that percentage increases to 20%, with the constraint of assigning containers in only two racks for each SH job, CPU utilization is lower in the HHN 75% solution.

Per-job metrics. Figure 5(a) shows the difference in job response time between the original Hadoop and HHN 75% for a range of 51 jobs (job 100 to job 150) in the trace. Most of the jobs with longer response times in HHN 75% than in original Hadoop were SH jobs. This is because containers in a maximum of two racks can be assigned to SH-job map tasks. On the other hand, this constraint sometimes helps the regular jobs that arrive near SH jobs to finish faster in HHN 75%, e.g., job 135 finishes 2.9 s earlier in HHN 75%. Job 134, which has the largest response time difference (i.e., 324 s), is the large SH job that caused the makespan difference discussed earlier.

Figure 5(b) shows per-job unfairness for the two solutions. Overall, the Hadoop fair scheduler with delay scheduling used in the EPS-only original Hadoop solution achieves better fairness. In the HHN 75% solution, the unfairness of large SH jobs is higher because these jobs are constrained to use containers in only two racks. Even though the modified YARN in the HHN solution offers SH jobs preferential treatment by allowing only SH jobs to place container requests in per-rack queues, SH jobs experience higher unfairness. Comparing Figs. 5(a) and 5(b), we observe a mirror-like pattern in the two metrics, i.e., unfairly treated SH jobs usually have longer completion times.

E. Sensitivity to System Parameters

We examined the impact of two system parameters on system and per-job metrics: (i) system size, and (ii) the percentage of ToR EPSs connected to OCS in the hybrid network. Two system sizes were used: four racks and 12 racks. Two values of the percentage of ToR EPSs connected to OCS were assumed: 75% and 100% (see Table III). These two cases are denoted by HHN 75% and HHN 100%. In HHN 100%, SH data sets are allowed to be stored on all racks, but the number of replicas per data set is still only two.

Two types of job traces were used: TS1 and TS2 (see "Workloads" paragraph of Section IV.B). Job arrival rate λ was increased in these runs relative to the values used in the runs described in Section IV.D. In selecting λ, we tried to make the CPU utilization in the EPS-only (original Hadoop) solution the same for the four-rack and 12-rack cases. For the TS1 trace, approximately the same CPU utilization was achieved with λ values of 0.3/s and 1.6/s for the four-rack and 12-rack cases, respectively. For TS2, these numbers were 0.25/s and 1.2/s for the four-rack and 12-rack cases, respectively.

System metrics. Table VI compares the system metrics, makespan, and CPU utilization, under different settings. First consider the values obtained for traces generated with the TS1 input. The effect of system size on makespan is as follows. The percentage difference in makespan between the original Hadoop and HHN solutions in

TABLE V
COMPARISON OF HHN WITH ORIGINAL HADOOP IN THE EPS-ONLY NETWORK FOR TS1 TRACES ON A 12-RACK SYSTEM

SH-Job Percentage p_s		5%			10%			20%		
Job Arrival Rate λ (/s)		0.3	0.6	0.9	0.3	0.6	0.9	0.3	0.6	0.9
Trace properties	Last-job arrival time (s)	3097	1767	1354	1579	890	688	754	438	332
	Number of jobs in trace	800	804	801	397	399	402	197	201	202
Makespan (s)	HHN 75%	3140.4	1810.4	1403.5	1644.1	1151.0	1092.0	1132.3	922.4	830.0
	Original Hadoop	3139.7	1809.8	1397.0	1622.2	954.8	1005.2	797.1	598.1	560.1
CPU utilization (%)	HHN 75%	28.4	50.0	63.3	29.2	48.0	57.3	29.9	46.3	60.7
	Original Hadoop	28.4	49.9	64.1	29.7	50.7	59.0	34.9	57.3	72.8

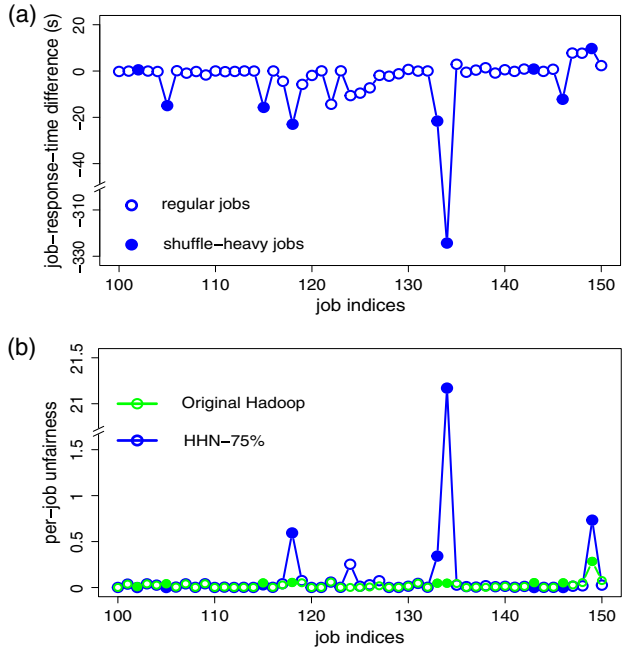

Fig. 5. Performance comparison of HHN 75% and original Hadoop in a 12-rack system; 50 jobs in a TS1 trace with $p_s = 20\%$, $\lambda = 0.6$ (view in color mode). (a) Job response time in original Hadoop minus that in HHN 75%. (b) Per-job unfairness.

a four-rack system was smaller than in the 12-rack system. This is because the time taken for all jobs to complete in the four-rack system provided more overlap of the large SH-job execution time with the execution times of other jobs. The effect of the percentage of ToR EPS connected to OCS in the hybrid network, 75% versus 100%, on makespan was not significant because the number of overlapping SH jobs was not high.

The CPU utilization in the 12-rack case is lower in the HHN solutions. This is because of the two-rack constraint on SH jobs.

Next, consider the results obtained with TS2 traces. Recall that SH jobs with more than 800-GB input data sets were excluded in TS2. The effects of the one large SH job (the input data set size for this job was 2.375 TB), which were described in Section IV.D, are not seen in the results for TS2. The makespan is almost the same in the original Hadoop and HHN solutions in both four-rack and 12-rack cases. Also, the CPU utilization is slightly better in the HHN solution.

We conclude that, for large input data sets, either more than two replicas should be created to spread out blocks on more racks (replicas of a block should necessarily be stored on different racks for reliability reasons), or even with just two replicas, the data sets should be spread out to more than two racks. However, the input data sets should not be so splintered between racks that the per-rack map output becomes too small to justify the use of optical circuits for shuffling.

Job response time. Figure 6 shows boxplots to compare job response times for various configurations.[1] The TS1 input was chosen, as it had worse results than TS2 (as shown in Table VI).

We make the following observations: (i) larger systems, i.e., systems with more racks outperform smaller systems (the job arrival rate λ values were chosen to make the CPU utilization the same in the four-rack and 12-racks cases for the original Hadoop configuration); (ii) in smaller systems, increasing the percentage of ToR EPS connected to OCS in the hybrid network affects regular jobs adversely, e.g., the job response time is longer for regular jobs in HHN 100% configuration than in the HHN 75% configuration, while the opposite is true for SH jobs.

Per-job unfairness. To gain better insight into per-job unfairness, we present this metric along with job response time for a particular setting: four-rack system, TS2 trace, $p_s = 20\%$, and $\lambda = 0.3$. Intuitively, if SH data sets are stored in all the racks of a hybrid network, regular jobs are likely to be treated unfairly because SH jobs receive preferential treatment with their use of per-rack queues. This effect should be more obvious in a system with a smaller number of racks because, with a larger number of racks, it is less likely for multiple shuffle-heavy jobs to be scheduled on all the racks at the same time. Thus, we choose the four-rack configuration to present the results.

Figure 7 presents the results. The original Hadoop on an EPS-only network offers the best fairness. Regular jobs suffer higher unfairness in HHN 100% when compared with HHN 75%. This is reversed for SH jobs.

This simulation run illustrates the effects on job response time of the system parameter, i.e., percentage of ToR EPS connected to OCS in the hybrid network. Therefore, Fig. 7(b) has been added to the job unfairness figure. It is apparent that SH jobs enjoy shorter response times in HHN 100% than in HHN 75% because all four racks in HHN 100% can be used to store SH data sets; hence, there is a smaller likelihood of multiple concurrent SH jobs competing for containers in the same two racks. With this trace, this exact scenario occurs when three consecutive SH jobs (job 35–37) have to share the same two racks based on the location of their data set replicas. Figure 7 shows that the unfairness level shoots up in the HHN 75% (blue) configuration for these three jobs; simultaneously, job response time increases.

Regular jobs enjoy the same short completion times in HHN 75% as they do in the original Hadoop on EPS network because there is always one rack out of the four racks that does not run SH jobs.

F. Sensitivity to a Hadoop Parameter

Here, we study the impact of one Hadoop parameter, i.e., number of replicas of each input block, on the system and

[1]One SH job in the four-rack instance of the HHN 75% configuration took 846.3 s. This point was dropped from the graph for better visualization of the differences.

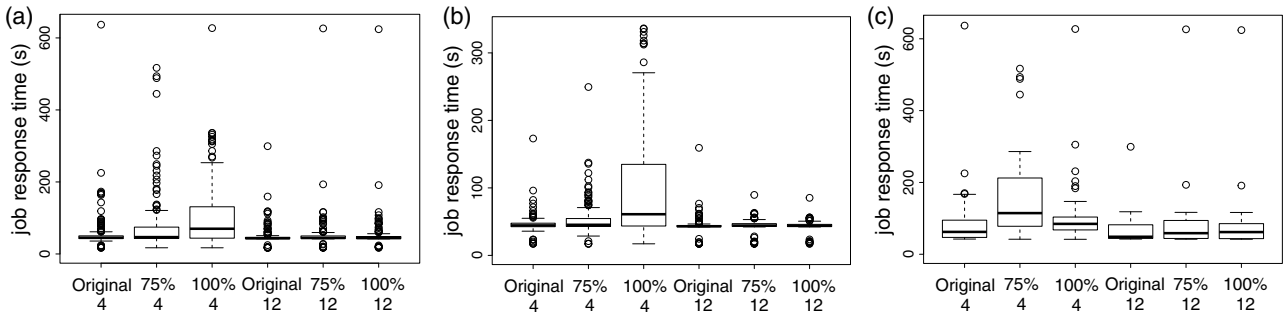

Fig. 6. Job response time comparison; four and 12, number of racks; original, original Hadoop on EPS-only network; 75% and 100%, HHN 75% and HHN 100%; TS1 input; $p_s = 20\%$; $\lambda = 0.3$ (four racks) and 1.6 (12 racks). (a) All jobs. (b) Regular jobs. (c) Shuffle-heavy jobs.

TABLE VI
COMPARISON OF SYSTEM METRICS IN TWO HHN CONFIGURATIONS AND ORIGINAL HADOOP ON EPS-ONLY NETWORK; $p_s = 20\%$

		4 Racks; $\lambda = 0.3$(TS1), 0.25(TS2)			12 Racks; $\lambda = 1.6$(TS1), 1.2(TS2)		
Trace	Metrics	HHN 75%	HHN 100%	Original Hadoop	HHN 75%	HHN 100%	Original Hadoop
TS1	Makespan (s)	1351.3	1346.6	1141.7	773.4	771.2	446.3
	CPU utilization (%)	80.7	80.1	82.7	72.8	72.4	83.9
TS2	Makespan (s)	1409.8	1403.3	1409.0	487.8	489.6	486.8
	CPU utilization (%)	83.0	82.7	82.4	81.2	80.8	80.4

per-job metrics. Two values of the number of replicas were used: 2 and 3 (see Table III). The traces used in this subsection are from TS1.

Job response time. Figures 8 and 9 illustrate the effect of the number of data set replicas on job response time in a four-rack system and a 12-rack system, respectively. We make the following observations: (i) in the smaller system with four racks, if only 75% of the ToR switches are connected to the OCS, and three replicas are used, response times for both regular jobs and SH jobs are statistically similar to the response times with the original Hadoop on an EPS-only DCN; and (ii) in the larger system

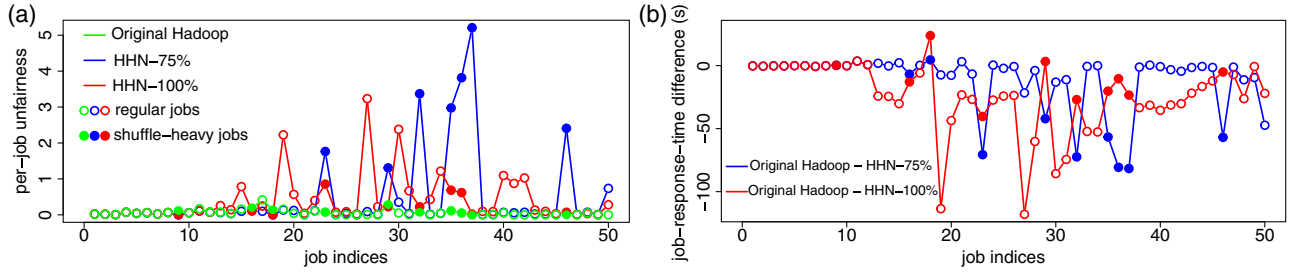

Fig. 7. Per-job metrics for the first 50 jobs of a TS2 trace; four-rack system; $p_s = 20\%$; $\lambda = 0.3$ (view in color mode). (a) Per-job unfairness. (b) Job response time diff. between original Hadoop and HHN 75%.

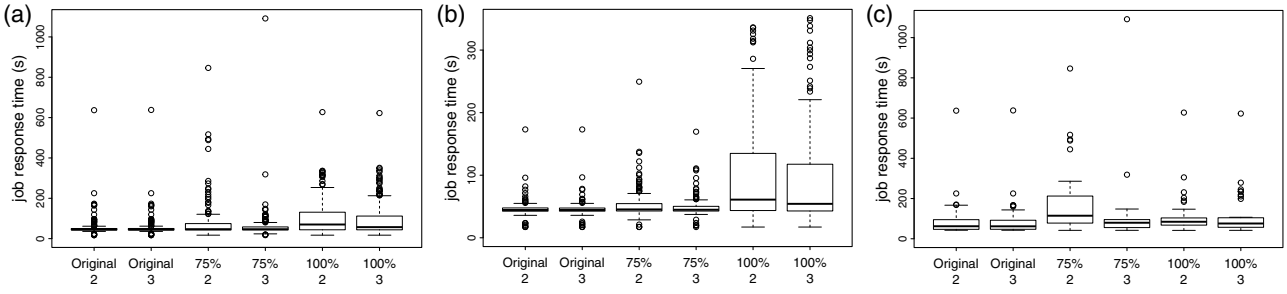

Fig. 8. Job response time comparison in a four-rack system; 2 and 3, number of replicas; original, original Hadoop on EPS-only network; 75% and 100%, HHN 75% and HHN 100%; TS1 input; $p_s = 20\%$; $\lambda = 0.3$. (a) All jobs. (b) Regular jobs. (c) Shuffle-heavy jobs.

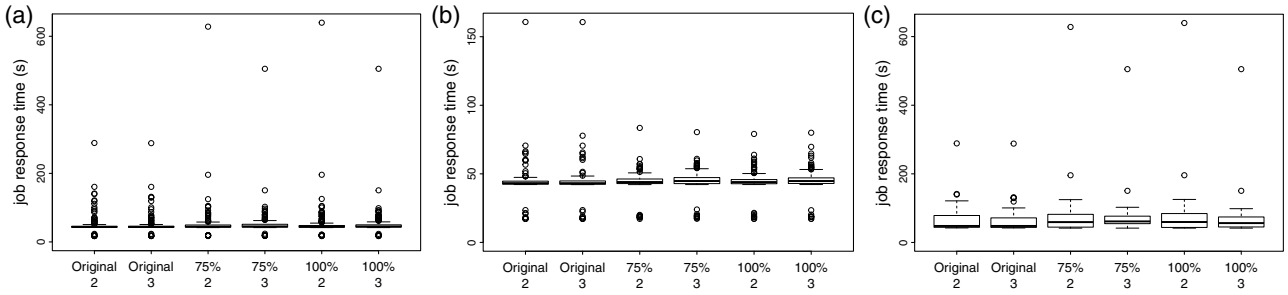

Fig. 9. Job response time comparison in a 12-rack system; 2 and 3, number of replicas; original, original Hadoop on EPS-only network; 75% and 100%, HHN 75% and HHN 100%; TS1 input; $p_s = 20\%$; $\lambda = 1.5$. (a) All jobs. (b) Regular jobs. (c) Shuffle-heavy jobs.

with 12 racks, SH-job response times are reduced when using three replicas when compared with the two-replica configuration, while regular-job response times are almost the same in all configurations.

Makespan. In Section IV.D, we observed that the makespan in HHN 75% is higher than that in the original Hadoop when the SH-job percentage is high (i.e., 20%). This is mainly due to the longer response time of a large SH job in HHN than in the original Hadoop. This occurs because, in HHN, each SH job can use containers in only two racks when the number of replicas is two. Because having three replicas helps to reduce response times of SH jobs in a 12-rack system (see Fig. 9), we expect it to also reduce makespan in HHN.

Table VII compares the makespan of TS1 traces in HHN and the original Hadoop under two settings for the number of replicas. When the SH-job percentage is small, i.e., 5%, the makespan in both HHN and the original Hadoop is not affected by the number of replicas. In contrast, the makespan in both HHN 75% and HHN 100% is reduced when having more replicas when 20% of the jobs are SH, while the makespan in the original Hadoop remains roughly the same for two and three replicas.

In summary, if the percentage of SH jobs is high, using a higher number of input-block replicas (e.g., 3) improves both job response time and makespan performance in HHN.

G. Multiple Traces With the Same Trace Parameters

The simulation results presented previously were all obtained using a single trace for each trace-parameter combination (SH-job percentage and job arrival rate). To test whether our conclusions were independent of the specific traces used, we generated multiple traces, i.e., 30 traces, for each trace-parameter pair (see Section IV.B). All the traces used in this subsection are from TS2.

We first compare the makespan performance for various configurations. Figure 10 shows boxplots of makespan for a 12-rack system at high load ($\lambda = 1.5$). Under both 5% and 20% SH-job scenarios [Fig. 10(a) and 10(b), respectively], the original Hadoop achieves shorter makespan than the two HHN configurations, but the difference is less significant when SH-job percentage is 5%. These observations are consistent with those made in Section IV.D. The longer makespan in HHN occurs because HHN limits SH jobs to run tasks on only a few racks even when there are idle containers in other racks. In addition, HHN 75% and HHN 100% have virtually the same performance.

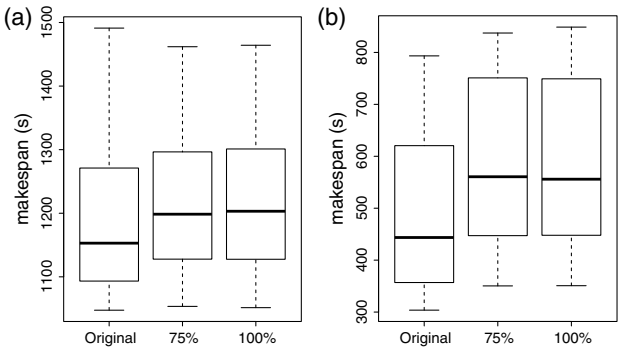

Fig. 10. Makespan comparison in a 12-rack system with different SH-job percentages; Original, original Hadoop on EPS-only network; 75% and 100%, HHN 75% and HHN 100%; TS2 input; $\lambda = 1.5$. (a) $p_s = 5\%$. (b) $p_s = 20\%$.

TABLE VII
MAKESPAN COMPARISON OF DIFFERENT NUMBER OF REPLICAS FOR TS1 TRACES IN A 12-RACK SYSTEM

SH-Job Percentage p_s	5%						20%					
Job Arrival Rate λ (/s)	0.3		0.6		0.9		0.3		0.6		0.9	
Number of replicas	2	3	2	3	2	3	2	3	2	3	2	3
HHN 75%	3140.4	3140.2	1810.4	1810.2	1403.5	1399.3	1132.3	1010.1	922.4	803.1	830.0	701.2
HHN 100%	3140.4	3140.2	1810.4	1810.2	1403.5	1402.4	1132.3	1010.1	922.4	803.1	830.0	699.5
Original Hadoop	3139.7	3140.1	1809.8	1810.0	1397.0	1397.1	797.1	796.8	598.1	599.2	560.1	560.0

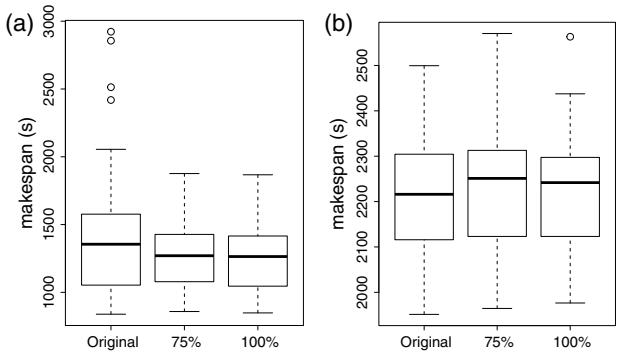

Fig. 11. Makespan comparison in a four-rack system at different loads; original, original Hadoop on EPS-only network; 75% and 100%, HHN 75% and HHN 100%; TS2 input; $p_s = 20\%$. (a) $\lambda = 0.3$. (b) $\lambda = 0.1$.

Figure 11 shows the makespan performance in a four-rack system. When the load is high ($\lambda = 0.3$), the original Hadoop ends up with much longer makespans than HHN for some traces. This is because, when the system is small, if there are several jobs with a large number of reduce tasks, most or even all of the containers could be allocated to those reduce tasks, leaving insufficient containers for map tasks. The system could enter a deadlock, i.e., reduce task wait for all map tasks to complete before starting execution, while map tasks wait for reduce tasks to complete in order to obtain containers to run. This problem is handled in the original Hadoop by preempting reduce containers, i.e., killing reduce tasks and allocating the freed containers to map tasks, which, however, leads to wasted CPU resources. The possible deadlock results in worse makespan performance for the original Hadoop than HHN. When the job arrival rate is lower ($\lambda = 0.1$), the original Hadoop on the EPS-only network works slightly better than HHN.

Next, we consider job response times. Figure 12 shows boxplots of the maximum response time of regular jobs and SH jobs in each of the 30 traces. For regular jobs in the four-rack system, the distribution of maximum job response time is further spread for the original Hadoop than HHN, which could be explained by the deadlock situation described above. The original Hadoop and HHN have similar distributions for maximum regular-job response time in the 12-rack system. For SH-jobs in the four-rack system, the maximum job response time is smaller for HHN than for the original Hadoop because

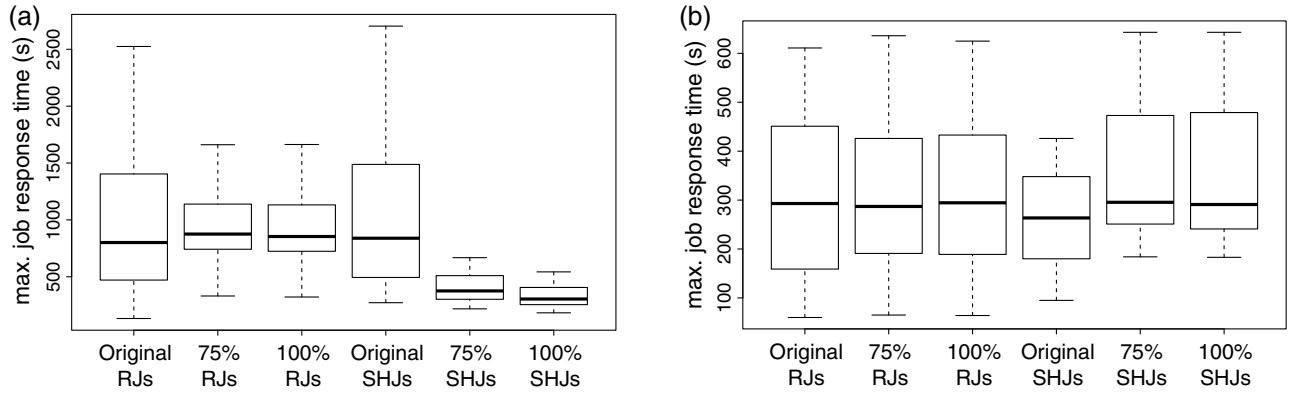

Fig. 12. Maximum job response time comparison; original, original Hadoop on EPS-only network; 75% and 100%, HHN 75% and HHN 100%; RJs, regular jobs; SHJs, shuffle-heavy jobs; TS2 input; $p_s = 20\%$. (a) four-rack system, $\lambda = 0.3$. (b) 12-rack system, $\lambda = 1.5$.

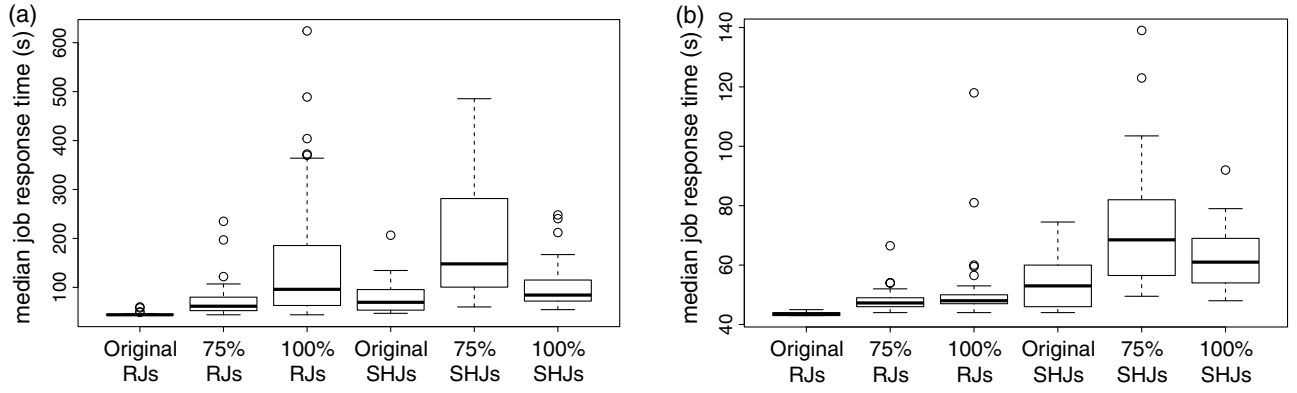

Fig. 13. Median job response time comparison; original, original Hadoop on EPS-only network; 75% and 100%, HHN 75% and HHN 100%; RJs, regular jobs; SHJs, shuffle-heavy jobs; TS2 input; $p_s = 20\%$. (a) Four-rack system, $\lambda = 0.3$. (b) 12-rack system, $\lambda = 1.5$.

the priority given to SH-jobs allows these jobs to obtain containers faster than they deserve in a fair scheduler. In contrast, large SH-jobs finish slower in HHN than in the original Hadoop in the 12-rack system. Although SH-job container requests are placed in per-rack queues, the map tasks of each SH-job are limited to use containers in only two racks. However, the worse performance of large SH jobs in HHN would be largely improved by using more input-block replicas (see Section IV.F).

As maximum job response time indicates the performance of very large jobs, we use median job response time to capture the performance of medium-sized jobs. There are two interesting findings in the median job response times shown in Fig. 13. The first is that, in both the four-rack and 12-rack systems, regular jobs perform worse in HHN 100% than in HHN 75%. This occurs because, when SH jobs are allowed to use all the racks in a system, regular jobs are likely to wait longer to obtain containers. The second finding is that, with both system sizes, medium-sized SH jobs finish slower in HHN 75% than in the original Hadoop and HHN 100%. This is because SH jobs need to wait in per-rack queues for service. The extra waiting time is longer in HHN 75% than in HHN 100%, and this waiting time is a larger portion of the response time for medium-sized jobs than for very large jobs.

V. Conclusions

The paper showed that it is feasible to modify certain data center applications so that the network traffic generated by these modified applications are better able to handle the high reconfiguration delays of OCS in hybrid electrical packet switch (EPS)/OCS networks. Specifically, this work proposed and evaluated a modified Hadoop designed for hybrid networks (HHN). Our evaluation results show that the HHN solution can achieve almost the same system-level performance metrics, makespan, and CPU utilization and per-job performance metrics such as response time and fairness, as the original Hadoop running on an EPS-only network with the same high-rate links as in the optical subsystem of the hybrid network. Because these high-rate links are always on in the EPS-only network and require more expensive high-speed transceivers in the EPS-only network, power consumption and costs are higher for the EPS-only network when compared with those of the hybrid network.

Acknowledgment

The University of Virginia portion of the work was supported by NSF grants CNS-1405171, CNS-1531065, CNS-1624676, and OAC-1659174.

References

[1] T. White, *Hadoop: The Definitive Guide*, O'Reilly Media, 2009.

[2] M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, and I. Stoica, "Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling," in *5th European Conf. on Computer Systems, EuroSys*, New York, New York, ACM, 2010, pp. 265–278.

[3] F. Ahmad, S. T. Chakradhar, A. Raghunathan, and T. N. Vijaykumar, "ShuffleWatcher: shuffle-aware scheduling in multi-tenant MapReduce clusters," in *USENIX Annual Technical Conf. (USENIX ATC)*, 2014, pp. 1–13.

[4] V. Jalaparti, P. Bodik, I. Menache, S. Rao, K. Makarychev, and M. Caesar, "Network-aware scheduling for data-parallel jobs: plan when you can," in *ACM Conf. on Special Interest Group on Data Communication, SIGCOMM*, New York, New York, ACM, 2015, pp. 407–420.

[5] N. Farrington, G. Porter, S. Radhakrishnan, H. H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen, and A. Vahdat, "Helios: A hybrid electrical/optical switch architecture for modular data centers," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, pp. 339–350, 2010.

[6] G. Wang, D. G. Andersen, M. Kaminsky, K. Papagiannaki, T. Ng, M. Kozuch, and M. Ryan, "c-Through: part-time optics in data centers," *ACM SIGCOMM*, vol. 40, no. 4, pp. 327–338, 2010.

[7] K. Chen, A. Singla, A. Singh, K. Ramachandran, L. Xu, Y. Zhang, X. Wen, and Y. Chen, "OSA: an optical switching architecture for data center networks with unprecedented flexibility," *IEEE/ACM Trans. Netw.*, vol. 22, no. 2, pp. 498–511, 2014.

[8] G. Porter, R. Strong, N. Farrington, A. Forencich, P. Chen-Sun, T. Rosing, Y. Fainman, G. Papen, and A. Vahdat, "Integrating microsecond circuit switching into the data center," *ACM SIGCOMM*, vol. 43, no. 4, pp. 447–458, 2013.

[9] H. Liu, F. Lu, A. Forencich, R. Kapoor, M. Tewari, G. M. Voelker, G. Papen, A. C. Snoeren, and G. Porter, "Circuit switching under the radar with REACToR," in *11th USENIX NSDI*, Seattle, Washington, USENIX Association, Apr. 2014, pp. 1–15.

[10] H. H. Bazzaz, M. Tewari, G. Wang, G. Porter, T. Ng, D. G. Andersen, M. Kaminsky, M. A. Kozuch, and A. Vahdat, "Switching the optical divide: fundamental challenges for hybrid electrical/optical datacenter networks," in *2nd ACM Symposium on Cloud Computing*, ACM, 2011, p. 30.

[11] X. Wang, M. Veeraraghavan, Z. Lin, and E. Oki, "Optical switch in the middle (OSM) architecture for DCNs with Hadoop adaptations," in *IEEE Int. Conf. on Communications*, May 2017.

[12] X. Wang and M. Veeraraghavan, "An evaluation study of a proposed Hadoop for hybrid networks (HHN)," in *IEEE Global Communications Conf. (GLOBECOM)*, Dec. 2017, pp. 1–7.

[13] "SWIM workload repository," [Online]. Available: https://github.com/SWIMProjectUCB/SWIM/wiki/Workloads-repository.

[14] N. Hamedazimi, Z. Qazi, H. Gupta, V. Sekar, S. R. Das, J. P. Longtin, H. Shah, and A. Tanwer, "Firefly: a reconfigurable wireless data center fabric using free-space optics," in *ACM Conf. on SIGCOMM*, New York, New York, ACM, 2014, pp. 319–330.

[15] Y. Cui, S. Xiao, X. Wang, Z. Yang, S. Yan, C. Zhu, X.-Y. Li, and N. Ge, "Diamond: nesting the data center network with wireless rings in 3-D space," *IEEE/ACM Trans. Netw.*, vol. 26, pp. 145–160, 2018.

[16] C. Zhang, F. Wu, X. Gao, and G. Chen, "Free talk in the air: a hierarchical topology for 60 GHz wireless data center networks," *IEEE/ACM Trans. Netw.*, vol. 25, pp. 3723–3737, 2017.

[17] M. Ghobadi, R. Mahajan, A. Phanishayee, N. Devanur, J. Kulkarni, G. Ranade, P.-A. Blanche, H. Rastegarfar,

M. Glick, and D. Kilper, "ProjecToR: agile reconfigurable data center interconnect," in *ACM SIGCOMM Conf.*, New York, New York, ACM, 2016, pp. 216–229.

[18] A. Yamashita, W. Muro, M. Hirono, T. Sato, S. Okamoto, N. Yamanaka, and M. Veeraraghavan, "Hadoop triggered opt/electrical data-center orchestration architecture for reducing power consumption," in *19th Int. Conf. on Transparent Optical Networks (ICTON)*, IEEE, 2017, pp. 1–4.

[19] J. Wang, D. Wang, M. Zhang, M. Qiu, and B. Guo, "Similarity-based node distance exploring and locality-aware shuffle optimization for Hadoop MapReduce," in *IEEE Int. Conf. on Smart Cloud (SmartCloud)*, IEEE, 2017, pp. 103–108.

[20] "Calient S series optical circuit switch," [Online]. Available: https://goo.gl/hc4DWj, price obtained through a conversation with a Calient salesperson.

[21] "Glimmerglass intelligent optical system 600," [Online]. Available: https://goo.gl/n1dByn.

[22] "Arista 7160 series and Arista 7280R series comparisons," [Online]. Available: https://www.arista.com/en/products, prices obtained through a conversation with Arista.

[23] "Cisco Nexus 7700 switches environment data sheet," [Online]. Available: https://goo.gl/Qpmqeq.

[24] "Juniper QFX10002 Ethernet switch data sheet," [Online]. Available: https://goo.gl/igaMvv.

[25] "Huawei CloudEngine 12800 series data center switches," [Online]. Available: https://goo.gl/555Ffs.

[26] "Arista 100G optics and cabling Q&A document," [Online]. Available: https://goo.gl/EpkTV3, price obtained through a conversation with Arista.

[27] "Cisco 100GBASE QSFP-100G modules data sheet," [Online]. Available: https://goo.gl/SPKH1r.

[28] "Fiber optic patch cables from fs.com," [Online]. Available: https://goo.gl/VCeMvt.

[29] "Rivanna: UVA HPC system," [Online]. Available: http://arcs.virginia.edu/rivanna.