Simplifying Optical DCN Fabrics with Blocking Space Switching and Wavelength-Constrained WDM

Konstantinos Kontodimas[1(✉)], Kostas Christodoulopoulos[2], and Emmanouel Varvarigos[1]

[1] School of Electrical & Computer Engineering, National Technical University of Athens, Athens, Greece
vmanos@central.ntua.gr
[2] Nokia Bell Labs, Stuttgart, Germany

Abstract. The introduction of all-optical switching in data center interconnection networks (DCN) is key to addressing some of the shortcomings of state-of-the-art electronic switched solutions. Limitations in the port count, reconfiguration speed and cost of optical switches, however, require novel optical switching and DCN designs. We present the concept of a simplified DCN fabric that relies on a lean optical switch design of limited but scalable functionality that offers high reconfiguration speeds, real-time scheduling, efficient control and low equipment cost. To achieve these objectives, the proposed architecture relaxes the usual non-blocking switching requirements but opts for switching modules that are constrained in terms of the achievable space and wavelength input-output configurations. We analytically compare the functionality and complexity of the simplified fabric with those of a non-blocking switch. We evaluate the throughput performance of the simplified DCN fabric and compare it to that of other fabrics with a different level of functionality and centralized control.

Keywords: Time-wavelength-space division multiplexing · Blocking design · Slotted and synchronous operation · Dynamic resource allocation, scheduling · Matrix decomposition

1 Introduction

The widespread availability of cloud applications to billions of users and the emergence of software-, platform- and infrastructure-as-a-service models rely on Data Centers (DCs). As traffic within a DC (east-west) is higher than incoming/outgoing (south-north) traffic [1], DC interconnection networks (DCN) play a crucial role to the overall DC performance. State-of-the-art DCNs are based on electronic switches connected in Fat-Tree topologies using optical fibers, with electro-opto-electrical transformation at each hop [2]. However, Fat-Trees tend to underutilize resources, require a large number of cables and switches, suffer from poor scalability and upgradability and exhibit high energy consumption [3].

A. Tzanakaki et al. (Eds.): ONDM 2019, LNCS 11616, pp. 286–298, 2020.
https://doi.org/10.1007/978-3-030-38085-4_25

The introduction of optical switching in DCN is key to resolving these shortcomings. Many recent works proposed hybrid electrical/optical switched DCN [4–13]. However, optical switches have high reconfiguration times, posing barriers in their applicability in DCNs.

The first barrier in using all-optical DCN comes from the cost and reconfiguration speed of (full) crossbar optical switches. The trade-off between the radix and reconfiguration speed is not adequate for large scale DCNs, and the switches are expensive.

The second barrier comes from the need to compute schedules to allocate the optical resources which is infeasible to perform optimally or even sub-optimally in real-time. NEPHELE [12] studied a distributed crossbar optical network fabric using WSSs interconnected in several wavelength-division multiplexing (WDM) fiber rings. This is promising, as reconfiguration speed is improved, but the architecture is still not scalable as creating long rings (which essentially replace the optical crossbars) induces losses and makes synchronization harder.

The third barrier comes from the control plane. Typically, centralized control is assumed in hybrid electrical/optical DCNs, following the SDN paradigm [6, 10, 12], where a central controller/scheduler collects monitored traffic and reconfigures the optical switches accordingly. Such closed-loop operation is inefficient, since the control plane induces high latency, requires tight synchronization of the optical switches with each other and with the scheduler, and has difficulties in following the DC traffic which is rather dynamic with time [3]. In [14], Patronas et al. showed that centralized scheduling calculations can be accelerated, but the signaling overhead identifying flows, monitoring, communicating configurations to the optical switches) remains hard to scale.

In this paper we investigate approaches for overcoming these shortcomings by departing from prior optical DCN architectures in the following ways:

(a) By designing custom lean but constrained (in their space- wavelength-time switching capabilities) optical switches, we call them SLIM, that use a small fixed set of switching states [15], supporting much fewer than the $(nW)!$ input-output configurations/mappings that are possible for fabrics that operate as distributed crossbars of n radix and W wavelengths. The custom switches use hard-wired interconnection mappings and the number of their internal switching components may vary. Because of their simplified internal design, they could render a great increase in terms of ports-speed capabilities. Moreover, the cost of these optical switches is expected to be substantially lower. The drawback is of course the limited (blocking) functionality of these SLIM switches, which has to be overcome through intelligent design of the scheduling architecture of the overall DCN.

(b) By reducing the level of centralized control with the development of appropriate algorithms to allocate the resources sub- but near-optimally and in real time. As the switches support only a few input-output matchings, scheduling complexity decreases. The freedom of non-blocking switches is not really needed and (b) it cannot be exploited in real-time because of the need to choose among too many configurations.

Deterministic round robin scheduling (RRS) is a scheduling policy that statically rotates though all source-destination assignments. For n radix, this policy supports $O(n)$ input-output matchings and is sufficient to serve well uniform traffic patterns and it doesn't require centralized control. However, traffic in DC is not generally uniform. For non-uniform traffic, we investigate the scheduling approach of an adaptive Birkhoff von Neumann decomposition method (referred to as A-BvN) that produces the schedules by taking into account the traffic demands. In terms of signaling and scheduling complexity, the level of centralized control required by A-BvN is lower than the case of full crossbar decomposition, since the complexity depends on the number of supported input-output matchings. In what follows we distinguish a DCN fabric/architecture as either non-blocking or blocking, depending on whether it can implement any one-to-one input-output configuration/permutation or it supports a reduced set of input-output matchings.

2 A Lean Non-crossbar Switch Design

2.1 Shift Shuffle

First of all, we define the *shift shuffle.* A k-shift shuffle $SS_k(n)$ is a static interconnection mapping between n input and n output ports, according to the function $SS_k(n) : i \rightarrow (i + k - 2) \bmod n + 1$, for $1 \leq i \leq n$ and $k \geq 1$. In particular, the mapping is a k-shift cyclic permutation.

2.2 The Selector and De-Selector Element

In [15] the authors proposed a switch design using a monolithic gang-switched module as its elementary building block. The gang-switched module is implemented with MEMS beam-steering micromirrors. In our work, we call a slightly modified version of this module as *Selector Element.* A Selector Element $SE(n, m)$ has n space inputs and $n \cdot m$ space outputs and can be set in one of m states. For state i, with $i = 1, 2, \ldots, m$, input signal j, is forwarded to output $(i - 1) \cdot n + j$. Conversely, we define a complementary module called *De-Selector Element.* A De-Selector Element $DSE(n, m)$ has $n \cdot m$ space inputs, n space outputs and can be set in one between m states. For state i with $1 \leq i \leq m$, input signal j with $1 \leq j \leq n \cdot m$ is forwarded to output $(i - 1) \cdot n + j \bmod m$. Since the implementation of (De-)Selector Elements is MEMS-based, the number of states is the number of tilting positions (i.e. m), which affects the reconfiguration speed. Note that each input of an $SE(n, m)$ (or $DSE(n, m)$) may carry W wavelengths, with all wavelengths of a (space) input carried to the same (space) output.

2.3 The SLIM Switch as a Concept

The concept of SLIM switch is based on the idea of *active* Selector Elements and *fixed* (hard-wired) k-shift shuffles. We differentiate from the work proposed by the authors of [15] by allowing WDM multiplexing and a variable number of Selector Elements, which can be considered as a generalization of the switch design of [15]. The number of Selector Elements is configurable, defining the *complexity* of the SLIM switch. A SLIM switch is defined by the tuple (M, C) and is a two-stage switch, consisting of $SE(M, M)$ modules in the input stage and $DSE(M, C)$ modules in the output stage, with C being the number of (D)SE modules of a stage (both input and output stages have the same number of (D)SEs). In other words, $N = M \cdot C$ is the total number of fiber (space) inputs to the switch, which are divided in C groups, each of M fibers, and each group forms the inputs of a $SE(M, M)$ module of stage. Each of the N outputs of the input stage are connected to all the inputs of the output stage through a fixed/hard-wired interconnection pattern.

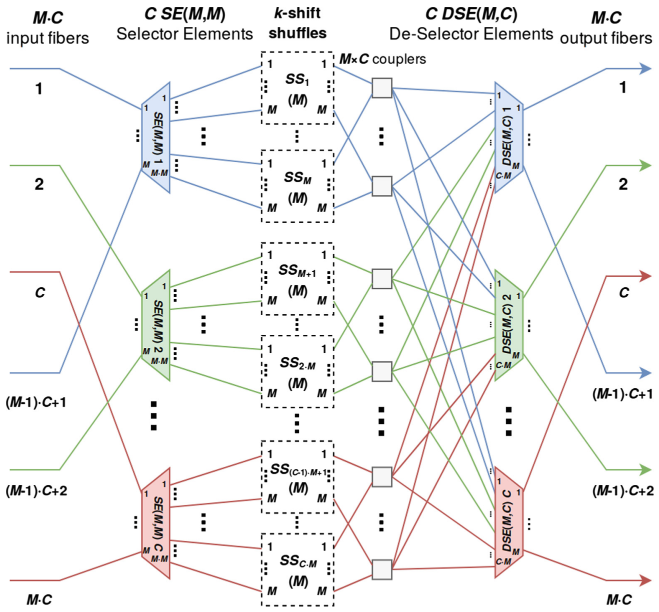

Fig. 1. The figure depicts a (M, C) SLIM switch. There is an input and an output stage: the input stage consists of $C\ SE(M, M)$ and the outputs stage consists of $DSE(M, C)$. There are M k-shift shuffles $SS_k(M)$ for $1 \leq k \leq M \cdot C$. The $M \cdot C$ input fibers are evenly distributed between the C SEs. Each SE of the first stage forwards M fibers to one between the $\cdot C\ SS_k(M)$, according to its state. The fibers are then combined according to the figure and broadcasted to all DSEs.

Each of the N outputs of the input stage are connected to $M \times C$ couplers through a fixed/hard-wired $\left(M^2 \cdot C\right) \times \left(M^2 \cdot C\right)$ [i.e., $(N \cdot M) \times (N \cdot M)$] interconnection pattern. Then the couplers are connected to the inputs of the output stage through a $\left(M \cdot C^2\right) \times \left(M \cdot C^2\right)$ [i.e., $(N \cdot C) \times (N \cdot C)$] interconnection pattern, as shown in general in Fig. 1. Each group k of M consecutive outputs of the SEs implements a distinct k-shift shuffle $SS_k(M)$. The fibers that are shuffled together are distributed between M couplers. Since only one group output of an SE contains active fibers, there are no conflicts in the couplers. Then, each coupler broadcasts its active signal to all DSEs. Depending on their states, the DSEs select to forward the signals originating by a particular SE.

3 A DCN Architecture with Non-crossbar Switches

3.1 The Architecture Specifications

The DCN model combines Electronic Packet Switching (EPS) and Optical Circuit Switching (OCS). The DCN architecture (Fig. 2) is organized in *racks/ToRs, PODs* and *SLIM switches*. Each rack hosts a ToR (top-of-rack) switch which is responsible for intra-rack and inter-rack communication of that particular rack. The ToR switch supports (a) EPS for intra-rack and (b) OCS for inter-rack communication using *tunable transmitters*.

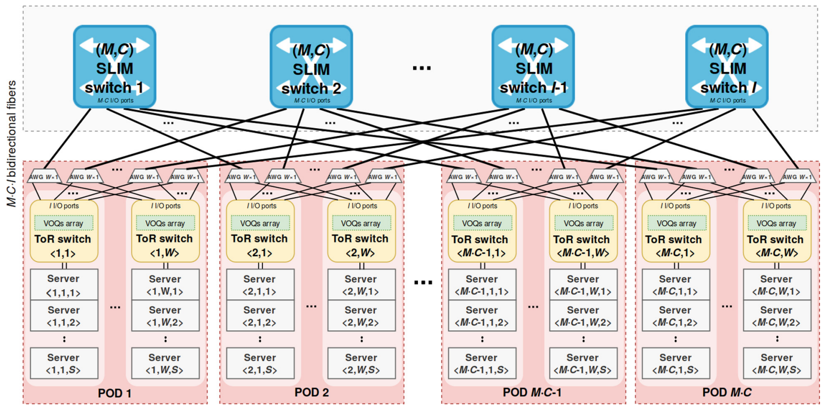

Fig. 2. There are $M \cdot C$ PODs, W racks/ToRs in each POD and W used wavelengths. The parameters M and C, also characterize the SLIM switches and they determine the number of switching patterns which are supported by the fabric. S is the number of servers/racks, I is the number of SLIM switches. Each server is identified by a $\langle p, w, s\rangle$ and each ToR/rack by a $\langle p, w\rangle$, where the p is the index of the POD, w is the index of a rack inside a POD and s is the index of a server inside a rack, with $1 \le p \le M \cdot C$, $1 \le w \le W$ and $1 \le s \le S$. The ToRs use I ports directed to the SLIM switches. All links imply bidirectional fibers. The servers in the same rack communicate with each other through EPS.

The racks are grouped in PODs (points of delivery). The racks in the same POD communicate with the racks of only one other POD, at the same time. The network carries WDM signal with wavelengths equal to the number of racks per POD. Each ToR listens to a wavelength, mitigating the conflicts among racks of the same POD. The tunable transmitters use the wavelength of the destination ToR. The ToRs belonging to the same POD (de)multiplex signals with arrayed waveguide gratings (AWGs). There are fibers for all switch-POD pair. A ToR switch dedicates an incoming and an outgoing port to a particular switch.

The racks communicate in a slotted manner, resembling the operation of a large TDMA switch with the ports of the ToRs being their input/output ports. The network maintains the timeslot component of TDMA, with the difference that slots are not statically assigned to circuits (ToR-to-ToR communications) but dynamically by a central scheduler according to traffic requirements. Deploying multiple SLIM switches extends the capacity. We generalize the timeslots to *generalized slots*.

Resource allocation is performed in *periods* of a number of timeslots; this enables important savings through the aggregation and suppression of control information and also helps absorb traffic peaks, reducing the dynamicity of resource allocation.

The parameters of a blocking DCN architecture are M, W, C, S, I and T. W is the number of racks/ToRs per POD, the number of wavelengths that the SLIM switch is able to carry, as long as the number of inputs (outputs) of the AWG (de)multiplexers. S is the number of servers per rack, I is the number of SLIM switches as long as the number of incoming/outgoing ports of the OCS interfaces of the ToR switches and their corresponding tunable transmitters. The number of PODs of the DCN is defined by the product $M \cdot C$. The factors M and C define the configuration of the SLIM switches. Finally, T is the number of timeslots of a period for which the scheduling decisions are made. Therefore, the number of racks/ToRs (size of the network) is $M \cdot W \cdot C$ racks, or equivalently $M \cdot W \cdot C \cdot S$ servers.

DCN's control is handled through a SDN-enabled *control plane*. The period is divided to three phases: *monitoring*, *scheduling* and *reconfiguration*. During monitoring phase, the control plane reads the traffic demands from the ToRs. Then, a batch scheduling is performed in order to decide which source-destination rack pairs take place at each generalized slot. In the last phase, the control plane disseminates the schedule to the switches. The ToRs use Virtual Output Queues (VOQs) in order to mitigate head-of-line blocking (HOL). The number of VOQs in each ToR switch is $M \cdot W \cdot C$. Using a SDN-enabled control plane also allows application-aware scheduling. Such option would further reduce the control cycle inefficiencies by eliminating the monitoring overhead altogether.

3.2 A Blocking DCN Fabric

The number of supported switching patterns by a switching fabric defines its *functionality*. In order to express a supported switching pattern by a blocking DCN fabric during a particular timeslot, we use matrix notation from linear algebra. In particular, it is expressed through a (M, W, C) *permutation matrix* (PM), defined as a $M \cdot W \cdot C \times M \cdot W \cdot C$ matrix $\mathbf{P} = \left[p_{ij}\right]$ with $1 \le i, j \le M \cdot W \cdot C$ and $p_{ij} \in \{0, 1\}$ that can be partitioned into $(M \cdot C)^2$ $W \times W$ submatrices (blocks) $\mathbf{P}_{rc}$, with $1 \le r, c \le M \cdot C$. r and c denote the indices of row and column partitions. A (M, W, C) PM satisfies the following constraints:

- C_1: Each row can have at most one entry set to '1'.
- C_2: Each column can have at most one entry set to '1'.
- C_3: If a submatrix is non-zero $\mathbf{P}_{rc}$, for $1 \le r, c \le M \cdot C$, a row partition $r' = (r + k \cdot C - 1) \bmod (M \cdot C) + 1$ can only have one non-zero submatrix in position $c' = (c + k \cdot C - 1) \bmod (M \cdot C) + 1$, for $1 \le k \le M$.
- C_4: If a submatrix $\mathbf{P}_{rc}$ is non-zero, for $1 \le r, c \le M \cdot C$, a column partition $c' = (c + k \cdot C - 1) \bmod (M \cdot C) + 1$ can only have one non-zero submatrix in position $r' = (r + k \cdot C - 1) \bmod (M \cdot C) + 1$, for $1 \le k \le M$.

- C_5: A submatrix $\mathbf{P}_{rc}$, for $1 \leq r, c \leq M \cdot C$, can have non-zero entries only in one diagonal between its main diagonal and the ones parallel to the main diagonal.
- A (M, W, C) PM generalizes the definition of permutation matrix. C_1 and C_2 are describe that a single source ToR can only be connected to at most one destination ToR vice versa. For C_3, C_4 and C_5 let us consider the following expressions. For $1 \leq r, c, r', c' \leq M \cdot C$:

$$r' = (r + k \cdot C - 1) \mod (M \cdot C) + 1 \tag{1}$$

$$c' = (c + l \cdot C - 1) \mod (M \cdot C) + 1 \tag{2}$$

Figure 3 in the next page, shows submatrix dependencies. C_5 forces the usage of only W cyclic matching between the ToRs of a given pair of PODs. This constraint doesn't reduce equipment of the DCN, but it reduces the scheduling complexity and the reconfiguration message overhead. There are various similar formal definitions regarding the distinction between a non-blocking and a blocking switching fabric. In our case, it is determined by whether there exist any switching patterns satisfying the trivial constraints C_1 and C_2 that cannot happen, or not.

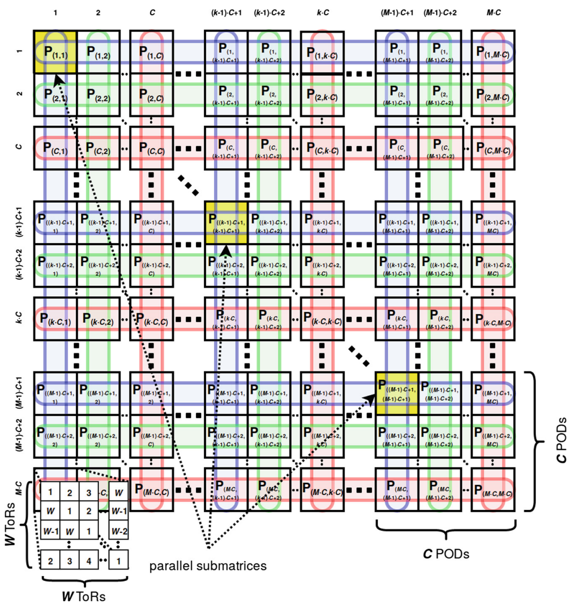

Fig. 3. The figure depicts a (M, W, C) permutation matrix, in submatrix level. The numbers index the row and column partitions of the matrix. Given a particular submatrix, all submatrices that belong to row (column) partitions that share the same color with the row (column) partitions of the former are *row-dependent* (*column-dependent*) to the former. The highlighted submatrices are examples of *parallel* submatrices. The inner entries submatrix $\mathbf{P}_{(M \cdot C, 1)}$ with the same number show which ones can mutually have non-zero entries.

For a network with size $M \cdot W \cdot C$ a non-blocking fabric supports all $(M \cdot W \cdot C)!$ switching patterns, deriving from all permutations of size $M \cdot W \cdot C$, while a blocking fabric supports less. The limited functionality of fabric applies when $W > 1$ or $M > 1$.

Theorem 1: A (M, C) SLIM switch carrying W wavelengths supports $C! \cdot \left(M \cdot W^M\right)^C$ switching patterns.

The proof is omitted due to space limitations.

4 Blocking VS Non-blocking Complexity

4.1 Control Message Complexity Comparison

The control message complexity is a factor that burdens the efficiency of a closed-loop non-blocking DCN. In a blocking fabric, at each period of T timeslots, the whole traffic matrix is required by the control plane, as in the case of I large crossbar switches of size $M \cdot W \cdot C$. Therefore, the monitoring complexity is $\Theta\left((M \cdot W \cdot C)^2\right)$. However, the complexity of the control messages for the reconfiguration phase is reduced. The non-blocking fabric induces a complexity of $O(T \cdot I \cdot M \cdot W \cdot C)$ due to the fact that each one of the source ToRs has to be informed for their scheduled destination ToR, for all non-zero generalized slots of a period which are upper-bounded by $T \cdot I$. On the other hand, the complexity for the messages in a blocking DCN is reduced to $O(T \cdot I \cdot M \cdot C)$. This is due to C_5 constraint, described in Sect. 3.2, which dictates only cyclic assignments of wavelengths to the ToRs that belong to the same POD. Therefore the wavelength assignment can be sufficiently described just by a scalar with a value ranging between 1 and W.

4.2 Scheduling Complexity Comparison

In this section we investigate the scheduling complexity. Let's use the notation of the *traffic matrices*. A traffic matrix $\mathbf{D} = \left[d_{ij}\right]$ is a $M \cdot C \cdot W \times M \cdot C \cdot W$ matrix, where each entry d_{ij} is the total number of *cells* that need to be transmitted from ToR switch i to ToR switch j. As cells, we define the normalized demands in bandwidth, with the respect to the capacity provided by a generalized slot. A traffic matrix can be partitioned into $(M \cdot C)^2$ $W \times W$ submatrices $\mathbf{D}_{r,c}$ for $1 \leq r, c \leq M \cdot C$. A traffic matrix is transferred to the control plane at each period consisting of T timeslots, describing the accumulated traffic of that period.

We define the *critical sum* h of the traffic matrix $\mathbf{D}$ as $h \triangleq \max\left\{\sum_{i'=1}^{M \cdot C \cdot W} d_{i'j}, \sum_{j'=1}^{M \cdot C \cdot W} d_{ij'}\right\}, \forall i, j$ where $1 \leq i, j \leq M \cdot C \cdot W$. According to a well-known theorem (shown in [17], p. 57), in order to decompose a traffic matrix $\mathbf{D}$ with critical sum h into a sum of permutation matrices, a scheduler would need to iteratively execute a maximum cardinality matching algorithm for h times. In a non-blocking fabric, the optimal scheduling would be handled through a Birkhoff von Neumann (BvN) decomposition algorithm. We refer to the BvN decomposition algorithm that is applied to a non-blocking fabric as EXACT [18]. Such decomposition gives a sequence of permutations of maximum cardinality, while the constraints described in Sect. 3.2 hold. Using the Hopcroft-Karp [16] algorithm, the complexity of finding a maximum matching for a dense matrix is $O\big((M \cdot W \cdot C)^{5/2}\big)$. We consider a traffic matrix as *admissible* if its critical sum is $h \le T \cdot I$. The traffic described by an admissible traffic matrix can be served with the capacity of $T \cdot I$ generalized slots of a period. For an admissible traffic matrix, the number of steps is upper-bounded by $T \cdot I$. Therefore, the worst-case complexity is $O\big(T \cdot I \cdot (M \cdot W \cdot C)^{5/2}\big)$.

In the case of a blocking DCN fabric that uses (M, C) SLIM switches with W wavelengths, the scheduling is reduced to a decomposition of an admissible traffic matrix into a sum of (M, W, C) PMs which follow the fabric's blocking constraints. The decomposition algorithm A-BvN is a generalization of a regular BvN decomposition algorithm.

Algorithm 1 Decomposition algorithm A-BvN

```
Step 1) Inputs. Inputs are the traffic matrix $\mathbf{D} = [\mathbf{D}_{rc}] = [d_{ij}]$ with
    $1 \le r,c \le M \cdot C$ and $1 \le i,j \le M \cdot W \cdot C$ and the scalars $M$, $W$, $C$, $T$ and $I$.
Step 2) Definitions. Define the matrices $\mathbf{A} = [a_{ij}] \in \mathbb{N}^{M\cdot C \times M\cdot C}$, $\mathbf{Q} = [q_{ij}] \in \mathbb{N}^{C \times M\cdot C}$,
    $\mathbf{Z} = [z_{ij}] \in \mathbb{N}^{C \times C}$, $\mathbf{Y} = [y_{ij}] \in \mathbb{N}^{C \times C}$, $\mathbf{S} = [s_{ij}] \in \mathbb{N}^{T\cdot I \times M\cdot W\cdot C}$, $\mathbf{X} = [x_{rcw}] \in \mathbb{N}^{M\cdot C \times M\cdot C \times W}$ and
    $\mathbf{e} = [e_i] \in \mathbb{N}^{C}$.
Step 3) Init. Set schedule matrix to zero $\mathbf{S} \leftarrow [0]^{T\cdot I \times M\cdot W\cdot C}$ and slot $t \leftarrow 0$.
Step 4) Find maximum cardinality diagonals. For each $r \in [1, M \cdot C]$,
    $c \in [1, M \cdot C]$ and $w \in [1, W]$: compute the cardinality of non-zero entries
    of the diagonal $w$, which is determined by the column index of the di-
    agonal's first row entry, of the submatrix $\mathbf{D}_{rc}$; store the computed
    cardinality to $x_{rcw}$ and save index $w$ of the maximum diagonal to $a_{rc}$.
Step 5) Compute sums of parallel matrices. For each $r' \in [1, C]$, $c \in [1, M \cdot C]$:
    sum $x_{rcw}$ with $r = (r' + k \cdot C - 1) \bmod (M \cdot C) + 1$ and $w = a_{rc}$ for all $k \in [1, M]$;
    store the sum to $q_{r'c}$.
Step 6) Find maximum between column-dependent matrices. For each $r \in [1, C]$
    and $c \in [1, C]$: find $c_{\max}$ such that $q_{rc_{\max}} \ge q_{rc'}$ with $c' = (c + k \cdot C - 1) \bmod (M \cdot$
    $C) + 1$ for all $k \in [1, M]$ and set $z_{rc} \leftarrow c_{\max}$ and $y_{rc} \leftarrow q_{rc_{\max}}$.
Step 7) Schedule. Set $t \leftarrow t + 1$.
Step 8) Run MWM. Consider the square matrix $\mathbf{Y}$ as a weighted bipartite
    graph, where $y_{ij}$ is the weight of the edge $(i,j)$. Run a Maximum Weighted
    Matching (MWM) algorithm with the square matrix $\mathbf{Y}$ as input. The out-
    put is a vector $\mathbf{e}$ with $e_r$, $1 \le r \le C$ having the column index $c$ with the
    maximum value.
Step 9) Loop over submatrices. For each $r' \in [1, C]$ and $k \in [1, M]$: set $c \leftarrow z_{r'c'}$,
    $r \leftarrow (r' + k \cdot C - 1) \bmod (M \cdot C) + 1$ and $l \leftarrow 0$ with $c' = e_{r'}$:
    Step 9a. Update the schedule. For each $i' \in [1, W]$ set $i \leftarrow (r - 1) \cdot W + i'$ and
        $j \leftarrow (c - 1) \cdot W + j'$ with $j' = (i' + a_{rc} - 2) \bmod W + 1$; if $d_{ij} > 0$ then set $s_{ti} \leftarrow j$
        and $l \leftarrow l + 1$, which is the count of covered non-zeros.
    Step 9b. Update maximum diagonal. Set $x_{rcw} \leftarrow x_{rcw} - l$ with $w = a_{rc}$. Find
        the new maximum diagonal among the already computed cardinalities
        $x_{rcw'}$ for all $w' \in [1, W]$ and store the diagonal index to $a_{rc}$.
    Step 9c. Update subsidiary matrices. Set $q_{r'c} \leftarrow q_{r'c} - l + x_{rcw}$ with $w = a_{rc}$.
    Step 9d. Update maximum between column-dependent matrices. Find $c_{\max}$
        such that $q_{r'c_{\max}} \ge q_{r'c''}$ with $c'' = (c' + k \cdot C - 1) \bmod (M \cdot C) + 1$ for all
        $k \in [1, M]$ and set $z_{r'c'} \leftarrow c_{\max}$ and $y_{r'c'} \leftarrow q_{r'c_{\max}}$.
Step 10) Loop. If there exist $i,j$ s.t. $d_{ij} > 0$ and $t < T$, then go to Step 7.
Step 11) Finish. Return the schedule $\mathbf{S}$.
```

Theorem 2: The worst case complexity of A-BvN is $O\big(T \cdot I \cdot (M \cdot W \cdot C)^2\big)$.

The proof is omitted due to space limitations.

4.3 Reconfiguration Speed Comparison

The reconfiguration speed depends on the number states of the SEs. All SEs have to tilt between all shift shuffles. For a DCN fabric based on (M, C) SLIM switches all C SEs need to tilt between $M \cdot C$ shift shuffles. The reduction of reconfiguration delay is derived from the decrease of the $M \cdot W \cdot C$ positions of a non-blocking fabric to $M \cdot C$, which is achieved through WDM. Thus, a (M, C) SLIM switch induces an increase in reconfiguration speed with the trade-off of limited switching flexibility. However, since the SEs don't tilt between consecutive shift shuffles, this reconfiguration speed applies to the worst case scenario. Therefore, the reconfiguration speed, in terms of asymptotic complexity, is $O(M \cdot C)$, which is lower than $O(M \cdot W \cdot C)$ of the case of a non-blocking fabric.

4.4 Cost Comparison

Another factor of inefficiency of the non-blocking fabric is the cost of its optical SEs. We compare the cost between the case of deploying a non-blocking DCN architecture and the case of deploying a DCN architecture based on SLIM the switch; the comparison takes place with respect to the number of (a) SEs used in each case and (b) the number of tunable transmitters. We consider the network parameters to be equal between the two DCNs. A MEMS-based non-blocking fabric consisting of I crossbar $M \cdot W \cdot C \times M \cdot W \cdot C$ switches would require $I \cdot M \cdot W \cdot C = \Theta(I \cdot M \cdot W \cdot C)$ MEMS in total. A blocking DCN fabric with I (M, C) SLIM switches requires a total number of $I \cdot C = \Theta(I \cdot C)$. However the blocking fabric uses WDM that requires the deployment of tunable transmitters at each ToR switch. The number of tunable transmitters deployed at each ToR is I, equal to the overall number of SLIM switches. Therefore, there is an additional cost for the deployment of $I \cdot M \cdot W \cdot C$ tunable transmitters for all $M \cdot W \cdot C$ ToRs. A non-blocking fabric, with features similar to the architecture we propose, wouldn't require the deployment of tunable transmitters in the ToR switches. However, for large radix the scenario of such a network is infeasible anyway, due to the rest of inefficiency factors.

Table 1. A summary of the complexity comparison.

	Non-blocking	Blocking
Reconfiguration Control	$O(T \cdot I \cdot M \cdot W \cdot C)$	$O(T \cdot I \cdot M \cdot C)$
Cost in MEMS	$\Theta(I \cdot M \cdot W \cdot C)$	$\Theta(I \cdot C)$
Cost in tunable Tx	–	$\Theta(I \cdot M \cdot W \cdot C)$
Reconfiguration speed	$O(M \cdot W \cdot C)$	$O(M \cdot C)$
Algorithm Complexity	$O(T \cdot I \cdot \left(M \cdot W \cdot C)^{5/2}\right)$	$O\left(T \cdot I \cdot (M \cdot W \cdot C)^2\right)$

5 Simulation Results

In this section, we present a set of simulation results, comparing the achievable throughput between three architecture scenarios with their corresponding algorithms. The first scenario applies to a blocking DCN fabric with a decentralized control plane (abbrev. as *decentralized scenario*). Since this architecture is traffic-agnostic, there is no need to implement a larger set than the trivial $M \cdot W \cdot C$ cyclic switching patterns. The second scenario applies to a blocking DCN fabric that deploys a centralized control plane (abbrev. as *semi-centralized scenario*). In fact, the semi-centralized scenario simulates the operation of a SLIM switch-enabled fabric. The last scenario applies to a fully non-blocking (crossbar-like) DCN fabric that deploys a centralized control plane (abbrev. as *centralized scenario*).

The scheduling policies that apply to each of the scenarios are RRS, A-BvN (Algorithm 1 of Sect. 4.2) and EXACT, for the decentralized, semi-centralized and centralized scenarios, respectively.

The flow-level simulation framework was implemented in Python 2.7. The network configuration parameters are $M \cdot C = W = T = I = 12$. The traffic is synthetic, generating at each cycle an admissible traffic matrix (according to the load ρ). The results are sampled for the load levels of $\rho = 10\%$, 50% and 90%. The level of load dynamicity is fixed to 1%, while the level of flow dynamicity is 0.1%. We also classify the traffic in two distinct flow densities, by randomly selecting a non-conflicting pattern of $M \cdot C$ *hotspots* of the size of PODs. The flows belonging to a hotspot have 100% density while the rest of the flows are characterized by 8% density. This configuration generates a spatially non-uniform and slowly changing traffic pattern that is suitable for exhibiting the adaptability of the semi-centralized fabric, depending on its complexity. The distinction in two levels of flow density is quite realistic, since the DCN traffic is naturally classified into intra-POD and intra-POD.

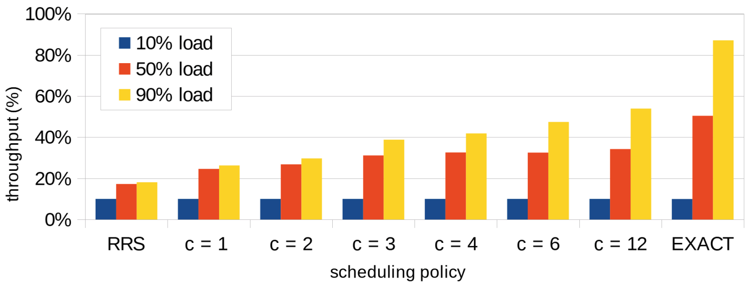

Fig. 4. The achieved throughput. RRS is used in the decentralized scenario while EXACT in the centralized. The semi-centralized scenarios in-between run A-BvN with complexities $C = 1$, 2, 3, 4, 6, 12.

It is clear by Fig. 4 that the decentralized scenario (RRS) is highly affected by the level of uniformity of the traffic pattern reaching a level of throughput of just 18%. The exact opposite is the case of the centralized scenario (EXACT) which reaches a maximum throughput of 87% for $\rho = 90\%$. The centralized scenario running EXACT is completely unaffected from the anomalies of the traffic as soon as the traffic matrix is admissible. However, the centralized scenario is not a sustainable option due to the inefficiencies presented in the previous sections. The in-between cases apply to semi-centralized cases under A-BvN. These cases achieve an adaptive level of throughput, namely 26% for $C = 1$, 30% for $C = 2$, 39% for $C = 3$, 42% for $C = 4$ and 47% for $C = 6$ and 54% for $C = 12$ (Table 1).

6 Conclusion

We presented a concept design of a blocking optical DCN fabric that copes with various inefficiencies of the operation of a fully non-blocking DCN architecture. The inefficiencies are caused due to high control overhead, high reconfiguration time in high radix and high deployment cost. This is achieved by utilizing the deployment of a blocking optical switch which supports a reduced set of hard-wired interconnection mappings and WDM. We analytically compared the complexity between the case of a blocking and a non-blocking fabric. The simplification causes reduced functionality which affects the throughput. We presented a flow-level evaluation, through simulations, that compares the achieved throughput between non-blocking and blocking fabric designs that deploy decentralized, semi-centralized and fully centralized control. The results show that for admissible synthetic traffic with long-term flows and a high degree of locality, the semi-centralized control has an advantage of at least 45% in comparison to the decentralized control. The throughput increases by utilizing a larger number of SEs. The difference between a blocking and the non-blocking architecture is not so high, considering the benefits of the reduced complexity.

Acknowledgment. This research is co-financed by Greece and the European Union (European Social Fund-ESF) through the Operational Programme «Human Resources Development, Education and Lifelong Learning» in the context of the project "Strengthening Human Resources Research Potential via Doctorate Research" (MIS-5000432), implemented by the State Scholarships Foundation (IKY).

References

1. Cisco Global Cloud Index: Forecast and Methodology, 2014–2019
2. Al-Fares, M., Loukissas, A., Vahdat, A.: A scalable, commodity data center network architecture. ACM SIGCOMM **38**(4), 63–74 (2008)
3. Roy, A., Zeng, H., Bagga, J., Porter, G., Snoeren, A.: Inside the social network's (Datacenter) network. ACM SIGCOMM **45**(4), 123–137 (2015)
4. Farrington, N., et al.: Helios: a hybrid electrical/optical switch architecture for modular data centers. ACM SIGCOMM **41**(4), 339–350 (2010)
5. Wang, G., et al.: c-through: part-time optics in data centers. ACM SIGCOMM **41**, 327–338 (2010)
6. Christodoulopoulos, K., Lugones, D., Katrinis, K., Ruffini, M., O'Mahony, D.: Performance evaluation of a hybrid optical/electrical interconnect. IEEE/OSA J. Lightw. Technol. **7**(3), 193–204 (2015)

7. Ben-Itzhak, Y., et al.: "C-share: optical circuits sharing for software-defined data-centers. arXiv preprint arXiv:1609.04521 (2016)
8. Singla, A., et al.: Proteus: a topology malleable data center network. ACM SIGCOMM Workshop on Hot Topics in Networks (2010)
9. Peng, S., et al.: Multi-tenant software-defined hybrid optical switched data centre. IEEE/OSA J. Lightw. Technol. **33**(15), 3224–3233 (2015)
10. Saridis, G., et al.: LIGHTNESS: a function-virtualizable software defined data center network with all-optical circuit/packet switching. IEEE/OSA J. Lightw. Technol. **34**(7), 1618–1627 (2016)
11. Porter, G., et al.: Integrating microsecond circuit switching into the data center. ACM SIGCOMM **43**, 447–458 (2013)
12. Bakopoulos, P., et al.: NEPHELE: an end-to-end scalable and dynamically reconfigurable optical architecture for application-aware SDN cloud datacenters. IEEE Commun. Mag. **56**(2), 178–188 (2018)
13. Calabretta, N., Miao, W.: Optical switching in data centers: architectures based on optical packet/burst switching. In: Testa, F., Pavesi, L. (eds.) Optical Switching in Next Generation Data Centers, pp. 45–69. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-61052-8_3
14. Patronas, I., Gkatzios, N., Kitsakis, V., Christodoulopoulos, K., Varvarigos, E., Reisis, D.: Scheduler Accelerator for TDMA Data Centers", Parallel, Distributed, and Network-Based Processing (2018)
15. Mellette, W.M., et al.: A scalable, partially configurable optical switch for data center networks. IEEE/OSA JLT **35**(2), 136–144 (2017)
16. Hopcroft, J.E., Karp, R.M.: An n^5/2 algorithm for maximum matchings in bipartite graphs. SIAM J. Comput. **2**(4), 225–231 (1973)
17. Ryser, H.J.: Combinatorial Mathematics (No. 14) Mathematical Association of America; distributed by Wiley New York (1963)
18. Inukai, T.: An efficient SS/TDMA time slot assignment algorithm. IEEE Trans. Commun. **27**(10), 1449–1455 (1979)