Collisions Free Scheduling in the NEPHELE Hybrid Electrical/Optical Datacenter Interconnect

[1]K. Christodoulopoulos, [2]K. Kontodimas, [2]A. Siokis, [3]K. Yiannopoulos, [1]E. Varvarigos
1: School of Electrical and Computer Engineering, National Technical University of Athens, Greece
2: Computer, Engineering and Informatics Department, University of Patras, Greece
3: Department of Telecommunications Science and Technology, University of Peloponnese, Greece

***Abstract*— The NEPHELE datacenter network is divided into pods/clusters of racks and relies on hybrid electro-optical top-of rack switches that access an all-optical network consisting of WDM rings. To enable dynamic and efficient sharing of the optical resources and a collision-free network operation, the NEPHELE network is designed to operate in a slotted manner with a software-defined-network (SDN) based control plane. We describe the NEPHELE resource allocation problem, consider the wavelength conflicts on the shared WDM rings, translate them into resource allocation constraints and investigate the effect of these constraints on network performance. To do so, we define the worst-case traffic pattern and perform simulations to evaluate performance for the average traffic. Finally, we propose a variation to the NEPHELE architecture that reduces the effects of these wavelength conflict constraints on performance.**

Keywords—slotted operation; time division multiplexing (TDM); wavelength division multiplexing (WDM); rings; resource allocation; scheduling; wavelenght collsions;

I. INTRODUCTION

The widespread availability of cloud applications, as well as the emergence of platform- and infrastructure-as-a-service models, has been facilitated by the existence of centralized computing infrastructures, typically referred to as Data Centers (DCs). DCs are comprised of a large number of servers grouped in racks. The performance of a DC is determined not only by its computing capacity but also by the capabilities and performance of its interconnection network. Given current trends, incoming and local traffic within the DC will continue to increase at high rates in the following years [1]. As a result, interconnection aspects will play a crucial role, and thus, high throughput, scalable and energy/cost efficient network architectures are required to fully harness the DC potential.

Currently, the state-of-the-art networks in DCs are based on electronic switching in fat-tree topologies [2]. The fat-tree approach tends to underutilize resources, requires a high cable count, suffers from poor scalability, and, finally, is not energy efficient [3]-[4]. Storage is typically accessed over a separate network and implemented following Fibre Channel protocol. Current trends include the disaggregation of computing, storage, and memory resources, the convergence of data and storage networks, and the use of application-driven networking, some of which do not seem to match well with the current architecture solutions. To cope with the shortcomings of fat-tree networks, hybrid interconnects consisting of an optical circuit switching and an electrical packet switching network have been proposed. There are many recent works [3] proposing such alternative DC architectures, in which the heavy and long-lived traffic is selectively routed over the circuit switched network, while the rest of the traffic goes through the packet switched network. However, the classification of traffic is quite difficult, while it has also been observed that long-lived and heavy flows do not comprise the typical case, and a high connectivity degree for the interconnection network is needed [4].

To enable dynamic and efficient sharing of the optical resources in a collision-free way, the NEPHELE interconnection architecture was introduced, based on the concept of a slotted optical DC interconnect. In the NEPHELE network, "slots" are time segments that can be accessed for rack-to-rack communication following the time division multiplexing (TDM) concept. "Slots" (and therefore network resources) can be assigned dynamically to communicating racks, and the NEPHELE approach can attain high utilization of the network capacity, leading to both energy and cost savings. Traffic is forwarded over multiple fiber rings that utilize multiple wavelengths, that is, they utilize wavelength division multiplexing (WDM) as a means of increasing the capacity. In particular, the NEPHELE network consists of independent planes, each employing several fiber rings. A ring and a slot on it are used for a *transparent* connection between two racks. In this combined TDM and WDM environment, slots need to be assigned so as to avoid the wavelength conflicts within the rings.

In this paper we describe the NEPHELE DC network and the resource allocation problem on it, and focus on the constraints posed by that architecture. We formally describe the wavelength conflicts within the shared WDM rings, translate them into resource allocation constraints and discuss their effect on performance. To do so, we define the worst-case traffic. We propose a variation to the NEPHELE architecture that reduces the effects of wavelength conflicts and perform simulations to evaluate performance for average traffic.

II. THE NEPHELE INTERCONNECT

NEPHELE is a hybrid electrical/optical interconnect, built out of the NEPHELE pod switches, top-of-rack (TOR) switches and network interface controllers (NIC). The NEPHELE network consists of P pods (clusters) of servers. A pod is a collection of W racks, each consisting of Z servers; a rack is accessed through its respective TOR switch. The TORs inside a pod and between the pods are interconnected through I parallel and identical *planes*, each of which consists of R unidirectional rings connecting the P pods. Each fiber ring carries WDM traffic comprising W wavelengths, propagating simultaneously in the same direction (unidirectional). So, a pod consists of W TOR switches and I POD switches that are interconnected in the following way: each TOR switch has I

northbound ports, each facing one of the I POD switches. A wavelength addresses a specific TOR inside a pod (the number W of wavelengths equals the number of TORs in a pod), while wavelengths are reused among the pods. The TOR switch is a hybrid electrical/optical switch: the electrical part of the TOR is connected "south" to Z·L_e server ports with "conventional" Ethernet links and to Z·L_o optical server ports and "north" to other TOR switches through the I planes of the all-optical NEPHELE pod network. Fig. 1a briefly describes the NEPHELE interconnect.

As mentioned, NEPHELE operates in a slotted manner, resembling the operation of a single (but huge) TDMA switch with the ports of the TOR being its input/output ports (I·P·W ports). The NEPHELE maintains the (time) slot component of TDMA, with the difference that slots are not statically assigned to circuits (TOR-to-TOR communications). Rather, slots are dynamically assigned by a central scheduler based on the respective traffic requirements. However, making scheduling decisions on a per-slot basis seems to be prohibitive, due to communication and processing latency limitations. Thus, it appears to be much more efficient to perform the resource allocation periodically, so that scheduling decisions are taken for periods of T slots; this enables important savings through the aggregation and suppression of monitoring and control information, and also helps absorb traffic peaks, reducing the required dynamicity of the resource allocation process.

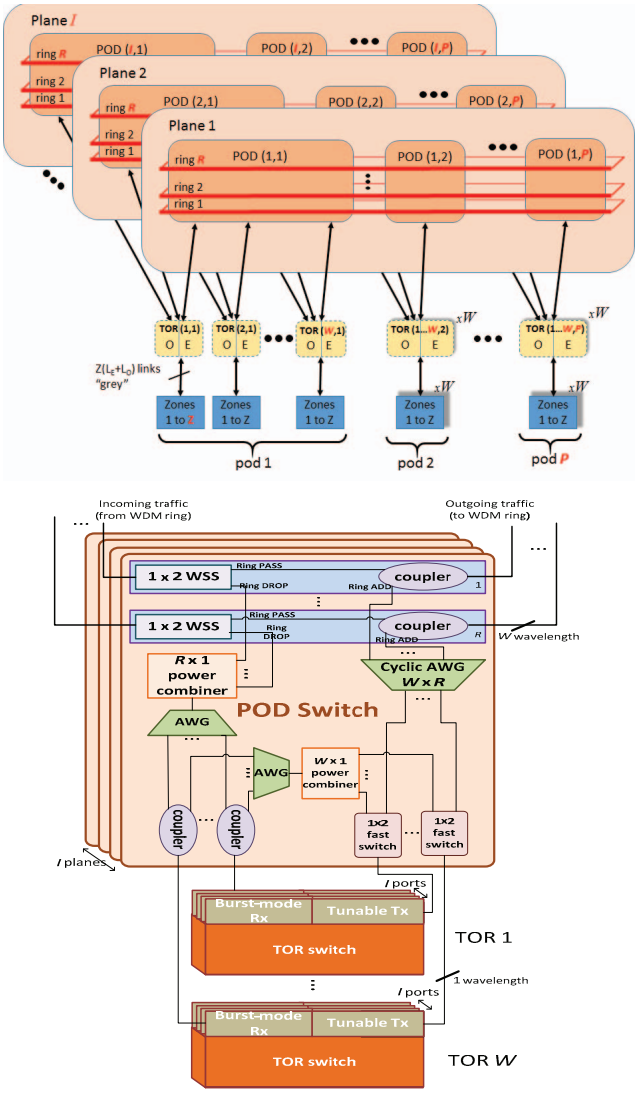

Fig. 1 (a) The NEPHELE architecture, (b) the NEPHELE POD switch

We now explain how communication is performed in the NEPHELE network assuming a slot as the switched entity (Fig. 1b). The traffic of the slot originating from a TOR enters a POD switch and is first switched through a fast 1×2 space switch according to locality; if the traffic is destined to a TOR inside the pod it remains within the POD switch, otherwise it is routed to the rings and to the next POD switch. Local intra-pod traffic, after the 1×2 switch, enters a $1\times W$ arrayed waveguide grating (AWG) where it is passively routed, depending on the wavelength of the signal and the input port. For a NxM cyclic AWG the static routing function that gives the output port is

$$out\_port=(in\_port+w) \bmod M,$$

where $0\le in\_port\le N$ is the input port, $0\le w\le$W the used wavelength, and 'mod' denotes the modulo operation. Inter-pod traffic is routed via the fast 1×2 switch towards a second $W\times R$ CAWG followed by couplers that combine multiple CAWG outputs into the fiber rings. So, the traffic enters the ring according to the CAWG function, propagates in the same ring through intermediate POD switches and is dropped at the destination pod. These routing decisions are applied by setting appropriately the related wavelength selective switches (WSS) in the POD switches. The WSSs can select whether a slot is passed or dropped on a per-fiber, per-wavelength and per-slot basis. Thus, each intermediate POD sets the related WSS to the pass state, while at the destination the related WSS is set to drop. The drop ports of the WSSs - corresponding to all the rings - are introduced into a $1\times W$ AWG and are passively routed to the W TORs in that pod. Through this AWG the traffic reaches the specific destination TOR.

III. NEPHELE Dynamic Bandwidth Allocation

NEPHELE aims to provide network resources in a dynamic and re-configurable fashion. To this end, TOR switches periodically report their bandwidth requests to the network controller, or applications report their requirements to the controller. The controller constructs a W·PxW·P traffic matrix (TM) at the end of each reporting period. Each traffic matrix entry TM(w_1,p_1,w_2,p_2) corresponds to the number of timeslots that have been requested for the communication between a source TOR(w_1,p_1) with a destination TOR(w_2,p_2), where p_1 and p_2 indicate the source and destination pods and w_1 and w_2 indicate the wavelength/position of the source and destination TORs inside the related pods, $1\le w_1,w_2\le W$ and $1\le p_1,p_2\le P$. This communication, however, must be performed in a coordinated manner to avoid collisions between transmissions inside the optical network. Let us denote an indicative communication as [TOR(w_1,p_1)→TOR(w_2,p_2),i,t], assuming that transmission takes place over plane $1\le i\le I$ at timeslot $1\le t\le T$. The following scheduling constraints (SC) must then be enforced:

SC1. No other communication [TOR(w_3,p_3)→TOR(w_2,p_2),i,t], for all $1\le w_3\le W$, $1\le p_3\le P$ on this slot t and plane i. This constraint ensures that the reception of the transmission is performed in a collision-free manner at TOR(w_2,p_2).

SC2. No other communication [TOR(w_1,p_1)→TOR(w_3,p_3),i,t], for all $1\le w_3\le W$, $1\le p_3\le P$ on this slot t and plane i. This constraint describes the fact that the TOR(w_1,p_1) cannot transmit to multiple destinations simultaneously in the space (plane) and time (timeslot) domains.

SC3. No communication [TOR(w_3,p_3)→TOR(w_2,p_4),i,t], for all $p_1<p_3<p_2$ or $p_1<p_4<p_2$, and $w_3=w_1+k\cdot R$, with k integer

so $1 \le w_3 \le W$, in the same slot t and plane i. This constraint ensures that no transmission that starts or ends at an intermediate pod and is forwarded to the same ring will appear within the same fiber ring (output of the CAWG).

The coordination of transmissions is achieved by expressing the TM as a sum of binary matrices, called *permutation matrices* (PM) that conform to all three constraints. Each PM represents the configuration of the network for a single slot and a single plane. Constraints 1 and 2 can be enforced in a straightforward way by allowing a single '1' per column and rowon the PM. The third constraint mandates that a single '1' will exist on the corresponding permutation matrix positions.

Note that in the above analysis, planes are identical and the scheduling conflicts are independent of the plane. Following this, slots and planes are treated homogeneously as a single type of resource to be allocated. The solution is written as $I \cdot T$ permutations, and there is no distinction regarding how these permutations are split into the related slots – plane resources.

The above bandwidth allocation problem becomes a TM decomposition problem into $I \cdot T$ permutations that can be optimally solved using the Birkhoff–von Neumann theorem [5] and bi-partite graph matching [6]. Given the high execution time of optimal decomposition, we have explored a heuristic scheduling algorithm that examines sequentially each TM entry and allocates an equal number of '1' in PMs by sequentially searching over them to find those that meet all three scheduling constraints. This algorithm exhibits improved execution time at the expense of underutilizing slots and planes, but this only manifests at relatively high network loads. Accounting for the typical (first two) scheduling constraints SC1 and SC2 is done in linear time, using data structures of size $I \cdot T \cdot P \cdot W$. Accounting for SC3 is done in superlinear time, using a data structure of size $I \cdot T \cdot P^2 \cdot W$ to keep track of the sub-rings that are defined by the source-destination pods of the connections (enabling the re-use of the remainder sub-ring by other connections). To reduce the running time we have considered a variation of *sub-ring* algorithm, referred to as *full-ring*, where a communication is assumed to conflict over the whole ring not the particular source-destination pod sub-ring (reducing to linear the size of the related data structure).

A. Worst case traffic

We now try to identify the worst case traffic pattern for a NEPHELE network that is built with I identical planes that have the above described scheduling constraints. The worst case traffic pattern assumes that all $Z \cdot (L_O + L_E)$ 'south' ports of TOR(w_1,p), $1 \le p \le P/2$, communicate with the respective 'south' ports located in TOR(w_2,$P-p+1$), and this traffic is bidirectional, meaning that we also have TOR(w_2, $P-p+1$)→ TOR(w_1, p) communication. All such communicating pairs, due to their specific TOR source and destination addresses, if routed on the same plane would result in the same ring using the same wavelength, that is, they are conflicting regarding scheduling constraint SC3. Assuming that a single server from each TOR communicates with a single destination server, there are $P/2$ connections conflicting (($P+1$)/2 if P is odd). Thus, $P/2$ planes are required for all such communication. Since there are $Z \cdot (L_E + L_O)$ servers/end ports in every rack, the total number of required planes is $I = Z \cdot (L_E + L_O) \cdot P/2$.

B. NEPHELE Architecture Variation

To reduce the number of planes required for the worst case traffic we proposed a variation of the reference NEPHELE architecture that employs non-homogeneous planes with different transfer functions. We assume that the drop AWG function of plane i, $1 \le i \le I$, and pod p, $1 \le p \le P$, is described by a constant $f(i,p)=c$, $0 \le c \le W-1$, which is different for different planes and pods (f=0 for all planes and pods in the baseline NEPHELE architecture). Following this assumption, a rack in a pod is no longer addressed with a specific wavelength, but the wavelength to reach it depends on the destination pod and the plane. To be more specific, consider again the communication TOR(w_1,p_1)→TOR(w_2,p_2) on plane i. Assume that we use wavelength w at the source so that ring $r=(w_1+w) \bmod R$ is used and $w_2=(w+f(i,p_2)) \bmod W$ is satisfied to reach the specific destination. Now, take a conflicting pair TOR(w_3,p_3)→TOR(w_4,p_4). In order for a pair to be conflicting, the same wavelength w and the same ring should be used, so $r=(w_3+w) \bmod R$ and also p_3 or p_4 should be in between p_1 and p_2. Thus, again $w_3=w_1+k \cdot R$ and $p_1<p_3<p_2$ or $p_1<p_4<p_2$ need to hold as in SC3. To reach the destination rack w_4 the following should hold: $w_4=(w+f(i,p_4)) \bmod W$. So in this case we do not need to have $w_4=w_2$ as was the case in SC3, but we need $w_4-w_2=f(i,p_4)-f(i,p_2)$. Our goal is to avoid this conflicting pair to conflict in any other plane. So we want to satisfy $f(i,p_4)-f(i,p_2) \neq f(i',p_4)-f(i',p_2)$, for all $1 \le i' < I$, $i' \neq i$. To be more general, we want the above to hold for any conflicting pair of any plane, that is for all p_1, p_2 ($p_2 \neq p_1$), p_3, p_4 ($p_4 \neq p_3$), w_1, w_2, and $i<I$.

For prime R we can construct a solution to this problem with the following recursive rules:

$$f(i,1)=1, \text{ for all planes } i,\ 1 \le i \le I,$$

$$f(i,p)=[f(i,j-1)+i-1] \bmod R+1,\ 1 \le i \le I,\ 2 \le p \le P.$$

For other R, such construction is not feasible, but still the agove solution is quite efficient. For prime R and the above rules the number of planes I required to serve any pattern is $I=Z \cdot (L_E+L_O)$. To see this, assume that all traffic is conflicting in a plane (as was the worst case traffic for baseline NEPHELE architecture described above). There are I-1 planes that this traffic is not conflicting, and assuming $P/2 \le I-1$, this traffic can be served by these planes. So, the number of planes required for worst case traffic and thus the number of 'northbound' TOR ports equals the 'southbound' TOR ports (to servers), typical for conventional TOR switches.

In this architecture variation, the wavelength used to reach each destination changes per plane. So, we need to have a map (can be pre-calculated) that would assign a wavelength to each combination of destination TOR per plane. This is actually a table lookup that takes sublinear time. Also, the transfer function of each plane is different, and thus, the planes are no longer equal and are not allocated the same as the slot resources. So, we need for each plane to pre-calculate the conflicting pairs and avoid those in the T slots that correspond to that plane. In this sense, the scheduling constraint SC3 translates into a plane assignment constraint. We can again have two variations, in the context of keeping track and examining the conflicts either in sub-rings or in full-rings; these are referred to as the *plane-variation/sub-ring greedy* and *plane-variation/full-ring greedy* algorithms, which require super-linear or linear time, respectively.

IV. Performance Results

We evaluate the performance of the NEPHELE architecture and the proposed variation via simulations for parameters: I=P=20 and W=T=80, which correspond to a fully-fledged NEPHELE implementation. The TMs were generated periodically (every T timeslots) with the adjustable traffic load and variation between successive TMs. In particular, we define (a) the network load ρ as the ratio of the total TOR traffic over the total capacity of a reporting period, (b) the densities d_{in} (d_{out}) as the ratio of the number of connections inside the pod (between pods) over the number of possible connections, (c) the load dynamicity parameter $\Delta\rho$ describing the percentage of change in the network load between successive periods, and (d) connection dynamicity parameter ΔS describing the average number of active connections that become inactivate within a period (an equal number of inactive connections activate on average, to keep the average density constant). Indirectly we define locality as the ratio of the load inside the pod to the total load: l=$d_{in}W$ / [$d_{in}W$ + $d_{in}W$(P-1)]. We assessed the performance of:

(i) reference architecture/greedy (without SC3)
(ii) reference architecture/ sub-ring greedy
(iii) reference architecture/full-ring greedy
(iv) plane variation/sub-ring greedy

Note that in all examined cases the number of planes was the same. Case (i) is used as a reference. The network architecture considered in case (i) can achieve maximum throughput, that is it can accommodate traffic of ρ=1. The network architecture of cases (ii) and (iii) has worst-case TM that require more (P/2 = 10 times) planes, while case (iv) also requires more planes than the I available, but lower than those of cases (ii) and (iii). The probability of generating the worst-case TM is extremely low, but cases (ii) and (iii) have several TM instances that require more than I planes, while for case (iv) this probability is negligible. Note, however, that we use a heuristic (greedy) and thus blocking is expected even for case (i).

Table 1 shows the maximum throughput for d_{in}=25%, Δp=1%, ΔS=1% and different values of d_{out} parameter. To find the throughput the non-served traffic of a period was added to the next period and we observed at which load traffic started to accumulate infinitely, typically referred to as unstable operation. Low locality (d_{out}=50%) results in heavy utilization of the inter-pod WDM rings and creates SC3 conflicts. We observe that for d_{out}=50% the throughput of the *reference architecture/sub-ring greedy* reduces to 0.5 as opposed to 0.92 of the *reference architecture/greedy* (*no SC3*), where we do not consider the effect of SC3. The *reference architecture/full-ring greedy* has even lower throughput (but at better execution times – as discussed next). The *different planes/full-ring greedy* resolves conflicts of one plane in another plane and thus improves substantially the throughput. The throughput was observed to be close to 0.85, very close to the case where we do not consider the SC3. As locality increases, inter-pod traffic reduces and eventually at high locality the performances of all the algorithms converge, as observed for d_{out}=0.5%. Note that it was observed that in a Facebook DC the locality is very high, higher than 50% [4]. Regarding the execution time (shown in Fig. 2 for d_{out}=12.5%), we observe that the *reference architecture/sub-ring greedy* algorithm has the highest running time, above 1 sec, since it keeps track of the utilization of the sub-rings which yields higher complexity. The *reference architecture/full-ring greedy* algorithm reduces the execution time (but wastes resources). The *reference architecture/full-ring greedy* algorithm has execution time slightly higher than the *reference architecture/greedy* (*no SC3*). The *different planes/full-ring greedy* has quite low execution time, similar to the *reference architecture/full-ring greedy*. So, it combines the benefits of the full ring algorithm in execution time and achieves throughput close to the case without SC3 constraint (reduces the conflicting sets).

TABLE I. Maximum Througput for Medium Dynamicity (1%)

Density between pods d_{out}	Locality	reference arch./ greedy (without SC3)	reference arch./ sub-ring SC3	reference arch./ full-ring SC3	plane variation/ sub-ring SC3
50%	2.5%	0.92	0.5	0.4	0.85
12.5%	10%	0.9	0.6	0.5	0.9
0.5%	70%	0.85	0.8	0.8	0.82

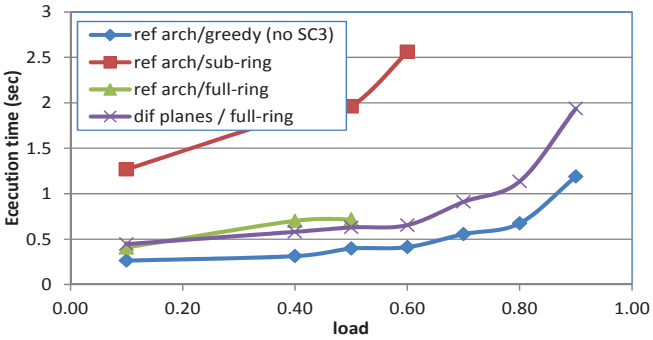

Fig. 2. Average execution time as a function of load for density between pods for d_{in}=25%, d_{out}=12.5% (locality 10%), Δp=1%, ΔS=1%.

V. Conclusions

The NEPHELE interconnect is based on dynamic allocation of slots to enable an efficient sharing of the optical resources and collision-free operation. We described the NEPHELE resource allocation problem and the architecture specific constraints. Wavelength conflicts on the shared WDM rings translate into resource allocation constraints that affect the performance. We identified the worst-case traffic and performed simulations to evaluate performance for the average traffic. We proposed a NEPHELE architecture variation that reduces the effects of the wavelength conflicts, with very low overhead.

Acknowledgment

Funded by E.C. through NEPHELE (grant agreement 645212)

References

[1] Cisco Global Cloud Index: Forecast and Methodology, 2014-2019

[2] Al-Fares, et. al.: A Scalable, Commodity Data Center Network Architecture, ACM SIGCOMM, 2008

[3] N. Farrington, et al.: Helios: a hybrid electrical/optical switch architecture for modular data centers. ACM SIGCOMM, 2010

[4] A. Roy, et. al., Inside the Social Network's (Datacenter) Network, ACM SIGCOMM, 2015

[5] G. Birkhoff, Tres observaciones sobre el algebra lineal, Univ. Nac. Tucuman Rev. Ser. A, 1946.

[6] J.E. Hopcroft, R.M. Karp, An n5/2 Algorithm for Maximum Matchings in Bipartite Graphs, SIAM Journal on Computing, 1973.