Bandwidth Allocation in the NEPHELE Hybrid Optical Interconnect

K. Christodoulopoulos[1], K. Kontodimas[2], K. Yiannopoulos[3], E. Varvarigos[1]
[1] *School of Electrical and Computer Engineering, National Technical University of Athens, Greece*
[2] *Department of Computer Engineering and Informatics, University of Patras, Greece*
[3] *Department of Informatics and Telecommunications, University of Peloponnese, Greece*
e-mail: vmanos@central.ntua.gr

ABSTRACT
The NEPHELE network is divided into pods of racks and relies on hybrid electro-optical top-of rack switches that access an all- optical network. To enable dynamic and efficient sharing of the optical resources and a collision-free network operation, the NEPHELE network is envisioned to be operated in a slotted TDMA manner with a software-defined-network (SDN) based control plane. We describe the NEPHELE resource allocation problem and present a number of algorithmic solutions suitable to tackle different traffic scenarios.
Keywords: Time division multiplexing, slotted – synchronous operation, dynamic resource allocation, scheduling, decomposition

1. INTRODUCTION

The widespread availability of cloud applications to billions of end-users, as well as the emergence of platform- and infrastructure-as-a-service models, have been enabled by the existence of centralized and optimized computing infrastructures (Data Centers - DCs). DCs are typically comprised of a large number of servers running Virtual Machines (VMs) that are grouped into communicating racks, and the DC performance is defined not only by its computing capacity but also by the capabilities and performance of its interconnection network infrastructure. Given the trend it becomes apparent that incoming and local traffic within the DC will continue to increase in the following years [1]. As a result, interconnection aspects of DCs will play a crucial role in their further development, 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]-[5]. 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 application driven networking that seem problematic under 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, such as [3]-[4], proposing such alternate DC architectures in which the heavy and long-lived traffic is selectively routed over the circuit switched network, while the rest 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 are not typical and a high connectivity degree is needed [5].

To enable dynamic and efficient sharing of the optical resources and a collision-free network operation, the NEPHELE architecture was proposed, introducing the concept of a slotted optical intra-DC interconnect design, where "slots" are time segments that can be accessed by a single rack-to-rack communication. "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. Moreover, multiple wavelengths and planes are utilized to implement a scalable and high capacity intra DC network infrastructure.

A brief description of the NEPHELE architecture is presented in Section 2, while in Section 3 we describe the dynamic resource allocation/timeslot assignment problem and present a set of algorithms to solve it. The paper concludes with a performance evaluation of the scheduling operation and the proposed algorithms.

2. NEPHELE hybrid optical/electrical interconnect

NEPHELE is a hybrid electrical/optical Data Center (DC) interconnect architecture, built out of pod switch, the NEPHELE top-of-rack (TOR) switch and the NEPHELE network interface controller (NIC) subsystems. In what follows we will refer to these subsystems as the TOR switch, POD switch, and NIC, respectively.

In NEPHELE interconnect there are I parallel planes, each of which consists of R *unidirectional* rings connecting P pods. Each one of the R fiber rings of the I planes carries WDM traffic comprising W wavelengths, propagating simultaneously in each ring in the same direction (unidirectional). A *pod* consists of I POD switches and W TOR switches that are interconnected in the following way: each TOR switch has I ports facing the I POD switches of the pod and, in particular, each port is connected to a different POD switch. The TOR switch is a hybrid electrical/optical switch and each one of the Z innovation zones (hosts, storage devices, memory or other systems) has L_E "conventional" Ethernet links to the electrical part of the TOR and L_O "all-optical" links from

NIC to the optical part of the TOR. So the TOR switch has in total $Z(L_E+L_O)$ ports facing 'south', that is facing the innovation zones. A diagram that briefly describes NEPHELE's interconnect is given in Figure 1. The actual selected values used in the techno-economic and performance evaluation studies are given in Table 1.

As it has already been mentioned, NEPHELE operates in a slotted TDMA manner, closely resembling the operation of a single (but huge) TDMA switch. In particular, NEPHELE maintains the *timeslot* component of TDMA, but timeslots are no longer statically assigned to circuits (TOR-to-TOR communications). Rather, timeslots are dynamically assigned to TOR-to-TOR communications by a central scheduler based on their respective traffic requirements. This approach enables a dynamic and optimal timeslot assignment and therefore complete utilization of the network resources. However, taking scheduling decisions in a per-timeslot basis seems to be prohibitive, due to communication and processing latency limitations. Therefore, instead of that it seems to be more efficient to do the resource allocation periodically so that scheduling decisions are taken for a timespan of *periods* or cycles of T timeslots; this way enables the aggregation and suppression of monitoring and control information and also absorbs traffic peaks, reducing the dynamicity of the resource allocation process.

Table 1. Reference network parameter values.

Parameter	Meaning	Target value
Z	Number of innovation zones per TOR	4
L_E	Number of "conventional" Ethernet links from Zone to TOR (EL Eth.)	4
L_O	Number of "all-optical" links from Zone to TOR (LO Eth.)	1
W	Number of TORs per POD; Also the number of wavelengths in the system	80
R	Number of rings per plane	20
P	Number of pods	20
I	Number of NEPHELE optical planes	20

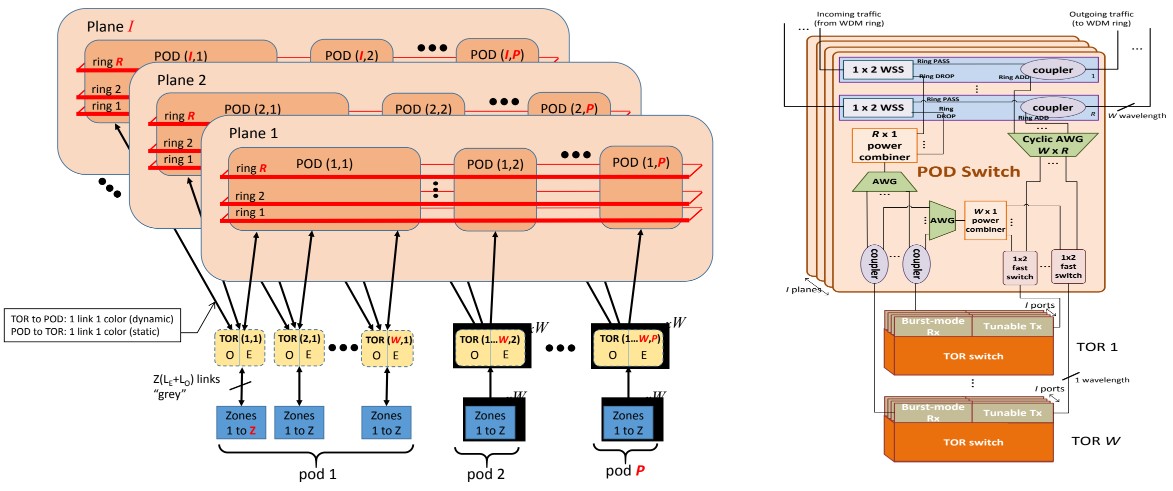

Figure 1. the NEPHELE interconnect.

3. 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 WPxWP traffic matrix (TM) at the end of each reporting period. Each traffic matrix entry TM(w_s,p_s,w_d,p_d) corresponds to the number of timeslots that have been requested for the communication between a source TOR(w_s,p_s) with a destination TOR(w_d,p_d). This communication, however, must be performed in a co-ordinated 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], that is the source is the w_1th TOR in pod p_1 and the destination is the w_2th TOR in pod p_2, $1\leq w_1,w_2\leq W$ and $1\leq p_1,p_2\leq P$, and the transmission takes place over plane i at timeslot t. The following scheduling constraints must then be observed:

1. We cannot have any other communication of the form [TOR(w_3,p_3)→TOR(w_2,p_2),i,t], for all $1\leq w_3\leq W$, $1\leq p_3\leq P$ 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).
2. We cannot have any other communication of the form [TOR(w_1,p_1)→TOR(w_3,p_3),i,t], for all $1\leq w_3\leq W$, $1\leq p_3\leq P$ 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.
3. We cannot have any communication of the form [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 $w_3=w_1+kR$, with k integer so that $1\leq w_3\leq W$, in the same slot t and plane i. This constraint

ensures that another transmission that starts or ends at an intermediate pod and is forwarded to the same ring will not appear on the same output of the cyclic AWG at the POD switch.

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 manner by allowing a single '1' per column and row, respectively, on the PM. The third constraint mandates that a single '1' will exist on the corresponding permutation matrix positions.

Following the above, dynamic bandwidth allocation becomes a TM decomposition problem that can be solved in an optimal manner using the Birkhoff–von Neumann theorem [6] and bi-partite graph matching [7]. Even though this approach ensures full utilization of the available timeslots and planes, the execution time of the optimal decomposition algorithm becomes prohibitive due to the extended number of interconnected PODs and TORs in NEPHELE, unless the TM is very sparse (i.e. hotspot TOR traffic dominates). Moreover, optimal solutions need to be updated at the end of each reporting period to take into account changes in the traffic demands of TORs and in the worst case scenario all permutations have to be re-computed anew.

Given the limited applicability of optimal decomposition, we have explored three heuristic decomposition algorithms that exhibit improved execution times at the expense of underutilizing timeslots and planes. As we demonstrate in the results section, there is a trade-off between resource utilization and execution time, but this only manifests at relatively high network loads. The proposed heuristics utilize previous decomposition solutions and appropriately modify them, given the updated traffic reports in the current period. This is performed in an *incremental* fashion and only traffic demands that have been modified are taken into account, while unaltered demands are served by existing permutation matrices. Consequently, the execution time of the heuristics is greatly improved and mostly depends on the traffic dynamicity (i.e. how fast the traffic pattern changes within a period) rather than the network size. In more detail:

1. Greedy heuristic: the greedy heuristic examines each TM entry and allocates an equal number of '1' in PMs where all three scheduling constraints are met. When seeking an incremental solution, a difference matrix (DM) between the current and previous TMs is constructed before executing the greedy heuristic. Negative DM entries, which correspond to reduced bandwidth requirements between the corresponding TORs or ceased connections, are initially set to '0' in existing PMs to facilitate new connections or connections with increased traffic. These connections manifest as positive DM entries and are allocated by sequentially examining existing PMs, again satisfying the three constraints.
2. Sublinear heuristic: the sublinear heuristic aims to reduce the number of sequential steps that are required for allocating traffic into PMs. This is accomplished noting that the TM matrix can be written as a sum of two matrices with (almost) equal entries, with each of the matrices being calculated by dividing the original TM by two. It is then only necessary to decompose one of the resulting matrices, since the solution for the other matrix is approximately the same, therefore reducing the execution time in half. Clearly, the execution time of the heuristic further improves, if larger divisors used to produce the matrices to decompose (i.e. when the TM is a sparser matrix).
3. Randomized heuristic: The randomized heuristic operates in a manner that resembles the greedy one, but allocates '1's in a randomized rather than sequential fashion to the PMs with a goal to increase the utilization of planes and timeslots on average. To this end, PMs that satisfy the constraints are first identified and a random assignment is then performed. In the incremental solution scenario, existing PMs are updated by randomly choosing '1's that correspond to negative DM entries and set them to '0'. PMs are then randomly augmented to serve traffic from increasing or new connections (positive DM entries).

4. Indicative performance results

The performance of the three heuristics was evaluated via simulations in *Matlab®* for a parameter set of $(I,P)=20$ and $(W,T)=80$, which corresponds to a fully-fledged NEPHELE implementation (see Table 1). The TMs were generated periodically (every T timeslots) with the adjustable traffic load and variation between successive TMs. In more detail, the traffic generator initially created a TM with identical row and column sums by randomly activating connections (TM entries). Each entry was taken to be equal to a parameter GI, which also reflects the sparsity of the TM: high sparsity (GI) values corresponded to the existence of traffic hotspots between TORs, since the number of non-zero TM entries equals $\rho \cdot I \cdot T/GI$, where ρ denotes the network load. Given an existing TM instance, the traffic generator would create a new one by randomly modifying the former by a selected percentage (0.1%, 1% or 10%) of the active connections, in order to take into account the impact of the temporal traffic dynamicity; these connections had their traffic reduced by GI, while other connections (active or inactive) had their traffic increased by the same amount, so that the total traffic transported in the network was constant among reporting periods and equal to $\rho \cdot I \cdot T \cdot W \cdot P$.

The heuristics were firstly assessed in terms of their average execution time over a time span of 1000 periods and the results are shown in Fig. 2 for a semi-sparse matrix (GI=40). It can be verified that the greedy and sublinear heuristics exhibit very fast execution times, in the order of tens of msec and are well-suited for bandwidth allocation in NEPHELE. The randomized heuristic is also viable in lower loads, while optimal decomposition typically takes tens of seconds to complete, thus it is not considered any further. Although the results presented in this work are limited due to the length limitation, similar measurements were taken for dense (*GI*=4) and sparse (*GI*=200) TMs. The results confirm the relative performance of the heuristics, although the execution times worsen as the density of the TMs increase, since the corresponding entries are more uniformly distributed and the satisfaction of the scheduling constraints becomes more challenging.

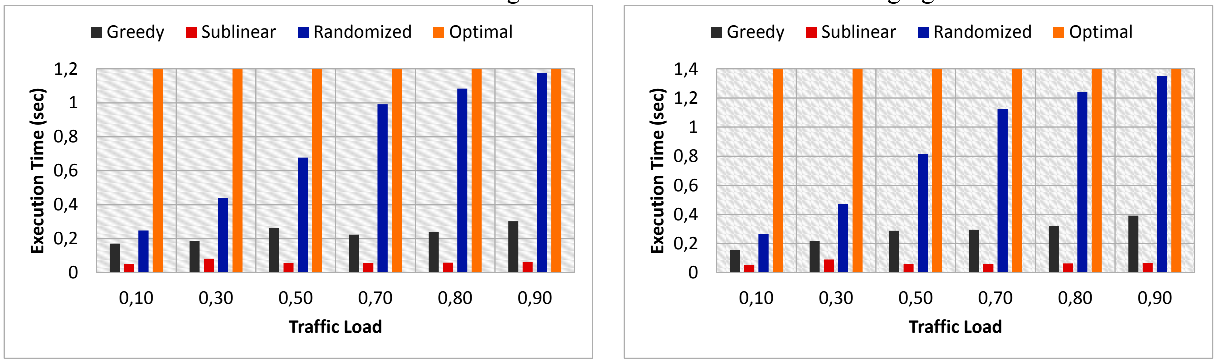

Fig. 2. Average execution time for a semi-sparse matrix (GI-40) with dynamicity equal to 0.1% (left) and 10% (right).

A second metric considered, is related to the resource utilization of each heuristic. To this end, the simulator monitored any traffic that was unserved within a period and added it to the TM of the next period. After a certain load, each heuristic started to uncontrollably accumulate traffic being unable to fully serve the incoming traffic, which is typically referred to as unstable operation. Table 2 details the load that drives each heuristic into instability. It can be seen that there is a tradeoff between the maximum traffic and the execution time; the greedy and randomized heuristics are more robust, while the sublinear is the most prone to blocking. Moreover, it is more often for blocking to appears in sparse matrices and this can be explained by the fact that the heuristics are more probable to satisfy the scheduling constraints when traffic is uniformly distributed (i.e. the TM is dense). It should be noted that the results of Fig. 2 and Table I were taken under scheduling constraints one and two only. The impact of the third constraint in terms of execution time is quite low, but it substantially increases the blocking performance; instabilities begin to occur at lower loads, for example for *GI*=40 and using the greedy heuristic, instabilities appear at loads 0.4, 0.5, and 0.5 for 0.1%, 1% and 10% dynamicity, respectively. Note that as dynamicity increases the blocking improves, since the traffic moves away faster from pathological patterns. In future, we target to look at multi-hop routing and alternative architectures that would relax the third constraint.

Table 2. Maximum (stable) load that can be served by each heuristic.

	Dynamicity 0.1%			**Dynamicity 1%**			**Dynamicity 10%**		
GI	Greedy	Sublinear	Randomized	Greedy	Sublinear	Randomized	Greedy	Sublinear	Randomized
4	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
40	0.8	0.7	0.8	0.8	0.8	0.8	0.8	0.8	0.8
200	0.6	0.6	0.7	0.7	0.6	0.7	0.7	0.6	0.7

5. CONCLUSIONS

We presented briefly the architecture of the NEPHELE hybrid optical/electrical interconnect that is envisioned to be operated in a slotted TDM manner to avoid collisions and achieve efficient resource utilization. We formally described the resource allocation problem and we proposed three algorithms to solve it. Our results showed a tradeoff between the maximum traffic served and the execution time of the algorithms; depending on traffic characteristics and system/control plane latency requirements the most suitable algorithm can be chosen.

ACKNOWLEDGEMENTS

This work was partially funded by the E.C. through Horizon the NEPHELE under grant agreement nº 645212.

REFERENCES

[1] Cisco Global Cloud Index: Forecast and Methodology, 2014–2019 White Paper
[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] G. Wang, et al.: c-through: part-time optics in data centers. ACM SIGCOMM, 2010
[5] A. Roy, et. al., Inside the Social Network's (Datacenter) Network, ACM SIGCOMM, 2015
[6] G. Birkhoff, Tres observaciones sobre el algebra lineal, *Univ. Nac. Tucuman Rev. Ser. A*, 1946.
[7] J.E. Hopcroft, R.M. Karp, An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs, *SIAM Journal on Computing*, 1973.