Traffic Prediction for Inter-Data Center Cross-Stratum Optimization Problems

Michał Aibin and Krzysztof Walkowiak
Department of Systems and Computer Networks
Faculty of Electronics
Wroclaw University of Science and Technology, Poland
Email: michal.aibin/krzysztof.walkowiak@pwr.edu.pl

Soroush Haeri and Ljiljana Trajković
School of Engineering Science
Simon Fraser University
Vancouver, British Columbia, Canada
Email: shaeri/ljilja@sfu.ca

***Abstract*—In this paper, we consider resource allocation in data networks and evaluate the performance of various approaches using the Cross-Stratum Optimization architecture for a provider-centric use case. We describe many approaches used to provision various requests related to a data center operations. We consider three simple approaches and compare them with two additional ones: the hybrid method that simultaneously considers cost, distance, and utilization; and traffic prediction methods based on Monte Carlo Tree Search that employs machine learning techniques. In evaluations, we rely on data center models and pricing structure provided by Amazon Web Services. Results indicate that using approaches that jointly optimize two strata improve network performance. Finally, the use of machine learning techniques enables network and data center operators to more efficiently utilize resources.**

***Index Terms*—software defined networks, elastic optical networks, data center services, dynamic routing, traffic prediction, machine learning, WAN**

I. Introduction

The rapid deployment of cloud computing and data center (DC) services with high bit-rate requirements substantially affects optical networks because majority of their traffic involves DCs. According to recent Cisco reports on network traffic ("Cisco Visual Networking Index" and "Cisco Global Cloud Index"), the annual global DC IP traffic will reach 15.3 zettabytes (ZB) in 2020, an increase from 4.7 ZB in 2015. DC to user traffic will grow 3-fold over the next 5 years at a compound annual growth rate of 27% from 2015 to 2020 while the DC to DC traffic will grow at the rate of 32% to almost 4-fold over the same period. The largest contributors are video-streaming services that deliver Internet video traffic to devices through a variety of Content Delivery Network (CDN) solutions. It is forecast that by 2020, 81% of Internet traffic will be delivered by CDNs, up from 51% in 2015. A significant increase of the DC traffic will result from the growing popularity of services such as: Internet of Things (IoT), consumer cloud storage, machine-to-machine communication, and Big Data applications.

In this paper, we consider Cross-Stratum Optimization (CSO) [1], [2] for provisioning of DC services in optical networks. It is expected that optical transport networks will evolve from the current wavelength-switched optical network (WSON) architectures built with the wavelength division multiplexing (WDM) technology towards elastic optical network (EON) architectures [3]. Flexible-grid EONs using a frequency grid with 2 x 6.25 GHz granularity (resulting in slots of 12.5 GHz) - in contrast to the 50 GHz fixed grid used in traditional WSONs - better utilize the spectrum resources [4], [5]. Consequently, EONs can provision variable bit-rate demands to adapt to the dynamic requests from DC services. Therefore, we consider here EON-based optical transport networks. The solutions and ideas proposed in this paper may be easily adapted to other optical approaches such as WDM technology.

The success of DCs has revolutionized the way various network services are delivered to end-users while bringing new challenges. In particular, the key goal is to achieve cost-effective and highly scalable implementations and deliver DC services with Quality of Service (QoS) guarantees using backbone optical networks. The strong competition among telecommunication companies in the global market has further influenced the need to develop appropriate and powerful approaches for these new services [6]. To answer this challenge, the concept of CSO has recently gained much attention. The key approach is to improve the performance of various network systems using a tighter coordination among network strata. CSO may allow global optimization and control across the optical transport network and the DC resources in order to meet the expected QoS requirements and to reduce the Capital Expenditure (CapEx) and Operating Expense (OpEx) associated with the provisioning of DC services.

Moreover, in order to provision a new lightpath for a DC request, manual operations are required, which precludes the establishment of dynamic connection requests. These drawbacks may be mitigated by employing the Software Defined Networking (SDN) concept. SDN solutions allow to decouple the control and data planes, centralize network intelligence, and abstract underlying network infrastructure from the applications [7]. Furthermore, SDN enables deployment of a centralized and programmable network control and management functions thus enabling efficient orchestration of network and DC resources as well as the application of various optimization techniques to control the network. Centralized network control becomes more important in hybrid and complex architectures such as CSO. The SDN controller logically centralizes the network intelligence and enables joint optimization of resources in two separate domains through the Open-Flow based

DC interconnection: the optical transport network (optical spectrum, network devices) and the DC (CPUs, memory, storage). Hence, enabling the network operators to coordinate DC service provisioning with the available optical network resources such as establishing lightpaths required to deliver the data between clients and DCs. As a consequence, SDN provides great flexibility for operators and significantly improves the overall system performance expressed by metrics such as CapEx/OpEx, throughput, and scalability.

The main contribution of this paper is an evaluation of a specific provider-centric use case for control and provisioning of DC services in WANs. We apply an advanced traffic prediction method based on the Monte Carlo Tree Search. This enables application of advanced optimization techniques for bandwidth-on-demand services and provisioning of dynamic lightpaths. We propose various optimization approaches that combine information about the network and DC resources available to the SDN framework. The approaches proposed in this paper do not depend on a particular implementation or architecture of SDN and, therefore, they apply to various scenarios.

The remainder of the paper is organized as follows. In Section 2, we formulate the considered problem and describe CSO architecture used in simulations. In Section 3, we present various approaches that may be applied to provisioning DC requests from the perspective of network operators, as well as we describe the traffic prediction mechanism. Simulation setup is described in Section 4. We present results in Section 5 and conclude with Section 6.

II. Problem Description

Over the past few years, a trend has emerged among many organizations, to use cloud computing facilities. One of the main reasons is the CapEx of creating or upgrading operating sites. Using cloud resources introduces following issues: identification DC base locations, performing network routing, and selecting locations for backups.

The current architecture may not provide the integrated end-to-end dynamic connectivity and high-level performance that cloud-oriented applications require. To allow optimization and control across optical networks and DCs, we consider the CSO architecture shown in Fig. 1.

A. Cross Stratum Optimization Architecture

The CSO architecture consists of the optical network where the EON technology is deployed, SDN controller that facilitates setting up and tearing down traffic flows, and DC infrastructure. The SDN obtains information about capacity and demand from the network devices installed in optical layer. It may, therefore, optimize flow management to provision DC requests and support service/user requirements for scalability and flexibility. Each DC has three resource types: computational (CPU units), RAM (GBs), and storage (TBs). Spectrum (THz) is the main considered resource in optical layer [4].

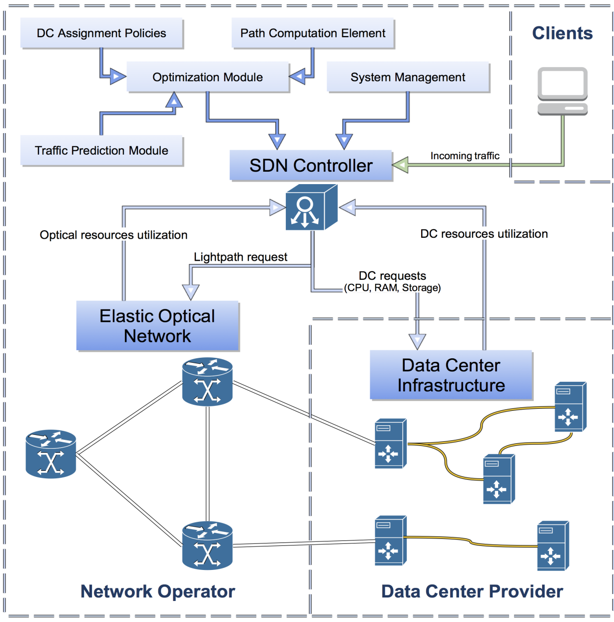

Fig. 1: Cross-Stratum Optimization (CSO) architecture for Data Center services provisioning. The architecture includes the network operator, DC provider, and clients.

The SDN controller has complete information about the current state of available resources to perform routing. Various approaches presented in this paper are implemented in the SDN controller in order to make the best decision for serving the incoming DC requests. The sequence of tasks is:

- New requests arrive in batches to the network. The SDN controller selects the best optical path and DC to serve the requests based on the implemented algorithms.
- The SDN controller sends information about the selected path and selected DC to the optical layer, which is responsible for decisions concerning the choice of modulation and allocation of link resources. Available modulation techniques are: Polarization Mode (PM)-BPSK, PM-QPSK, PM-8-QAM, PM-16-QAM, PM-32-QAM, and PM-64 QAM with spectral efficiencies of 1, ..., 6 (bits/s)/Hz, respectively. Note that the PM doubles the initial spectral efficiency. EON transmitter then selects spectrum range (slices) to establish the connection.
- After setting up connections, the SDN controller updates all network metrics.

The CSO architecture enables dynamic topology control with various implementations of the SDN controller such as a single controller, multiple controllers connected in a mesh, or hierarchical settings [8]. Moreover, all these implementations enable the usage of traffic prediction. This significantly enhances utilization of network resources and enables defining cost plans that are devised based on requested services.

For more details on the SDN orchestration for multi domain networks, please refer to [8].

III. Approaches for Provisioning Data Center Services

Three simple approaches that may be employed to identify a DC for providing services to a request are: selecting the nearest (to the request source), the cheapest, or the least utilized DC. While implementing these approaches is rather simple, preliminary simulation results show that they do not scale well and, hence, result in high ($\geq 1\%$) blocking percentage of requests even in networks with low to moderate traffic loads. We propose two approaches: a Hybrid DC assignment approach that combines the three simple approaches to improve network performance as well as a Traffic Prediction algorithm that employs the Monte Carlo Tree Search [9].

A. The Hybrid Data Center Assignment

We consider DC utilization and cost as well as the optical layer utilization of candidate paths from the request source to DCs to calculate a preference score for every (DC, candidate path) pair. The DC and the path with the highest score are then selected for service provisioning.

Let us assume that there are k DCs in the network. We consider n candidate paths between a DC and the request source. Utilization and price scores are first assigned to each of the k DCs. Utilization score of a DC is calculated as the ratio of the lowest utilization (CPU, RAM, storage) among all DCs in the network to the utilization of the evaluated DC. Similarly, the price score of a DC is the ratio of the cost of the cheapest DC among all DCs in the network to the cost of the evaluated DC. Hence, the least utilized and the cheapest DCs have the highest utilization and price scores, respectively. For the $n \times k$ candidate paths from the request source to DCs, utilization scores are then calculated as the ratio of the utilization of the least utilized path in the network to the utilization of the evaluated path. The preference score of a (DC, candidate path) pair is finally calculated as the sum of the DC utilization, price, and the candidate path utilization scores. In the Hybrid approach, we then select for service provisioning the (DC, candidate path) with the highest preference score.

B. Data Center Assignment with Traffic Prediction

In this approach, the DC requests are processed in batches. Monte Carlo Tree Search is then used to identify the best combination of (DCs, candidate path) pairs for provisioning service to the current batch of requests.

MCTS [10] is a promising approach for searching game trees for rewarding actions. If the decision-making agent has access to a generative model of the system that is capable of generating samples of successor states ζ' and rewards ι given a state ζ and an action a, it may be used to perform a sampling-based look-ahead search for rewarding actions [11]. Monte Carlo Tree Search builds a sparse search tree and selects actions using Monte Carlo sampling. These actions are used to deepen the tree in the most promising direction [12]–[14].

The nodes and edges of the search tree corresponding to states and actions, respectively. The root of the tree corresponds to the initial state ζ_0. Let $|A_\zeta|$ be the number of available actions at a given state ζ. The search tree node that corresponds to this state has $|A_\zeta|$ child nodes, each corresponding to a possible next state ζ' that is a result of selecting an action $a \in A_\zeta$. Each tree node stores a *value* κ and a *visit count* σ. A path from the root to a leaf node defines an action policy π.

The higher value of κ the better the quality of the decision. The κ is calculated using the trade-off between the cost of service and request blocking percentage. The MCTS balances path and DC utilization. For example it would be better to use paths and DCs that are moderately utilized than the non-utilized paths and highly utilized DCs. The σ indicates how often the particular pair of path and DC was chosen in the prediction. Choosing often the same DC and path pair, leads to higher utilization of it. The σ parameter balances the choices, by lowering the reward of DC and path pairs that are overused. The total reward is a sum of scores, calculated from the root to the termination leaf node, using the (2) equation. The goal is to choose the decision that leads to the higher value of the reward.

MCTS begins with a tree that only consists of the root node. It then executes until a predefined computational budget β is exhausted. In simple words β indicates the number of search tree levels that are going to be created.

The following four phases of the MCTS algorithm can be distinguished:

1) *Selection*: At this stage, the tree is traversed from the root until a non-terminal leaf node is reached. At each level of the tree, a child node is selected based on a *selection strategy*, which may be exploratory or exploitative. An exploratory strategy probes the undiscovered sections of the search tree in order to find better solutions. On the other hand, the exploitative strategy focuses on the promising subtrees that have already been discovered. The *exploration vs. exploitation* trade-off [15] must be considered when employing a selection strategy [16].
Furthermore, the Upper Confidence Bounds for Trees (UCT) [10] is used as a selection strategy. It is one of the most common selection strategies for MCTS algorithm. Let θ denote the visit count of current node of the search tree and Ψ the set of all its children. Furthermore, let κ_ψ and σ_ψ denote the value and visit count of a node with an index ψ. UCT selects a child χ from:

$$\chi = \arg\max_{\psi \in \Psi} \left(\frac{\kappa_\psi}{\sigma_\psi} + EX\sqrt{\frac{\ln\theta}{\sigma_\psi}}\right), \tag{1}$$

where EX is an exploration constant that determines the balance between exploration and exploitation. If $EX = 0$, the selection strategy is strictly exploitative.
The MCTS algorithm was originally introduced in Go game [16]. It is a two-player board game with three possible outcomes: win, draw, or loss. The solutions may be encoded as 1, 0, or -1, respectively. The UCT selection strategy (1) does not consider possible

deviations in the values of the children nodes. This deviation does not play an important role in two-players games. The deviation becomes more important in single-player games. It is similar to the problem in this paper. Single-Player MCTS (2) is a variant of MCTS that has been proposed for solving single-player puzzles [17]. It introduces a *deviation* term to UCT (1), hence:

$$\chi = \arg\max_{\psi \in \Psi} \left(\frac{\kappa_\psi}{\sigma_\psi} + EX\sqrt{\frac{\ln\theta}{\sigma_\psi}} + \sqrt{\frac{\sum_{\psi \in \Psi} \iota_\psi^2 - \sigma_\psi \kappa_\psi + LPC}{\sigma_\psi}} \right) \tag{2}$$

is used as the selection strategy, where ι_ψ^2 is the sum of the squared rewards that the ψ^{th} child node has received so far and LPC is a large positive constant.

2) *Expansion*: After a non-terminal leaf node is selected, one or more of its successors are added to the tree, to expand it. In this dissertation, the most common *expansion strategy* is used, which adds one node for every execution of the four MCTS phases. The new node corresponds to the next state of the prediction [16].
3) *Simulation*: From the given state of the non-terminal node that has been selected, a sequence of actions is performed until a terminal state is reached. Even though MCTS converges with randomly selected actions [10], utilizing domain knowledge may improve the convergence time [12].
4) *Backpropagation*: After reaching the simulation time limit or the terminal state, a reward is calculated. This reward is then propagated from the terminal node to the root in order to calculate the solution quality.

The computational budget β may be defined as the number of evaluated action samples per selection cycle or by selecting the time limit. In this paper, the five selection cycles for each set of requests was established. After repeating the four phases of MCTS β times, the child of the root with the highest average value is selected as an optimal action. The final result is a trade-off between the cost and the blocking percentage of the request. The MCTS then enters its next state and the selected child is chosen to be the new root of the search tree.

A search tree is first constructed where the root corresponds to the current DC and the optical resource utilization in the network. The root has $|R| \times k$ children for each (DC, candidate path) pair available for serving the current DC request. Monte Carlo simulations are executed using the current distribution of the DC requests to deepen the search tree up to β levels. When a leaf node at depth β is reached, its value is calculated as the sum of all utilization scores of DCs and optical links in the network. The (DC, candidate path) pair that corresponds to the root's child with the highest value is then selected for serving the current request. The runtime of the algorithm can be simply be computed as O($|A_\zeta| \times \beta$), where $|A_\zeta|$ is the number of random children to consider per search and β is the computational budget.

The advantage of this approach is that, time permitting, it may search for better optical path and DC assignments thus more efficiently utilizing the network resources. Network operators may adjust the execution time of this approach based on the traffic load in order to improve network performance.

IV. Simulation Setup

We have considered the Euro28 network (28 nodes, 82 unidirectional links, and 7 DCs) and US26 network (26 nodes, 84 unidirectional links, and 10 DCs). The location of DCs, interconnection points, and submarine cable landing stations are obtained from the *Data Center Map* website (`http://www.datacentermap.com`). In each DC location, ten $m3.2xlarge$ Amazon Web Services (AWS) EC2 instances are available. The pricing model for DC resources is based on the AWS [18]. The EON technology is used for the optical layer. In all simulation scenarios, the entire band of 4 THz spectrum, which is divided into 6.25 GHz frequency slices, is available thus resulting into 640 slices. Each network has three interconnection points to other networks that carry international traffic. We take into consideration the physical impairment of links (fiber attenuation, component insertion loss) and use regenerators for signals that require higher modulation formats. The traffic model is based on the 2018 projection of "Cisco Visual Networking Index" and "Cisco Global Cloud Index" reported forecasts. The types of requests presented in these reports are:

- *Processing as a Cloud (PaaC)*: Cloud providers deliver a computing platform, typically including operating system, programming-language execution environment, database, and web server. *PaaC* requests require CPU and RAM from DCs. We consider node to DC and international traffic to DC *PaaC* requests.
- *Storage as a Cloud (SaaC)*: This type of requests require storage resources from DCs. Cloud storage may be used for copying virtual machine images from the cloud to on-premises locations, to import a virtual machine image from an on-premises location to the cloud image library, or to move virtual machine images between user accounts or between data centers. We consider node to DC, DC to DC, and international traffic to DC *SaaC* requests.
- *Software as a Service (SaaS)*: This is one of the most popular types of cloud services sometimes referred to as "on-demand software". It is often priced on a pay-per-use or subscription basis. *SaaS* requires all three types of DC resources (CPU, RAM, storage). We consider node to DC and international traffic to DC *SaaS* requests.
- *Optical as a Service (OaaS)*: It requires storage resources from DCs and optical network resources in the network to transfer large amounts of data between nodes and DCs or between DC and DC.

We assume that the requests arrive in batches based on a Poisson distribution with the mean arrival rate of λ requests per unit time while their lifetime is exponentially distributed with the mean $1/\gamma$. Hence, the traffic load is λ/γ Erlangs. The number of requests in the simulation scenarios in both Euro28 and US26 is 457,000. We do not consider the first

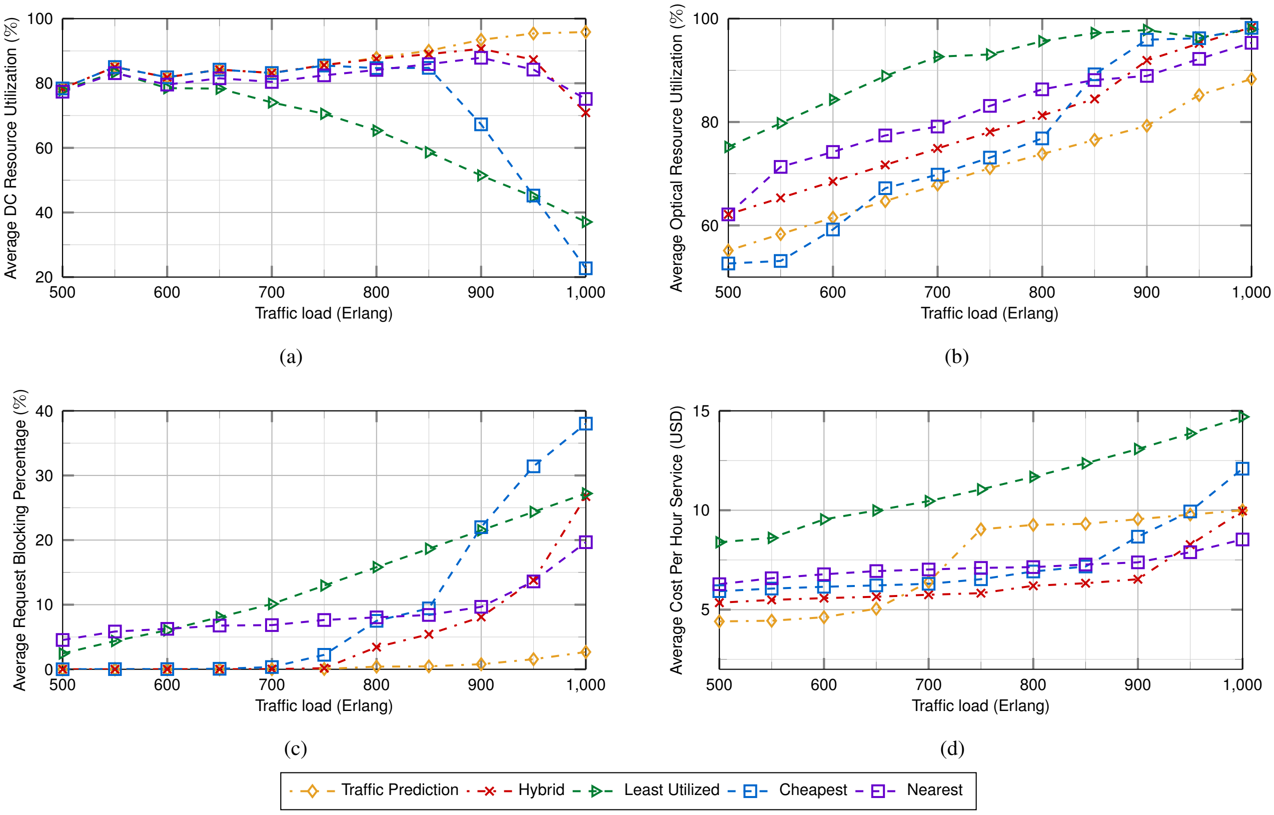

Fig. 2: Performance of the algorithms using the Euro28 network topology.

7,000 requests because the network loads did not reach a steady-state.

V. Simulation Results

In this Section, we present experimental results and compare performance of the proposed DC requests provisioning approaches. Results of Euro28 and US26 scenarios are shown in Fig. 2 and Fig. 3, respectively. We calculated average:

1) DC resource utilization (see Fig. 2(a)),
2) Optical resource utilization (see Fig. 2(b)),
3) Request blocking percentage (see Fig. 2(c) and Fig. 3), as a volume of rejected traffic divided by the volume of the entire traffic offered to the network,
4) Cost per hour of service (see Fig. 2(d)) of the entire network.

We first analyze the Euro28 network. Hybrid and Traffic Prediction approaches outperform the remaining three approaches as shown in Fig. 2. For lower traffic loads (less than 700 Erlangs), Traffic Prediction, Hybrid, and Cheapest approaches offer satisfactory performance ($\leq$ 1% blocking percentage) as shown in Fig. 2(c).

The Least Utilized DC approach rejects approximately 5% requests for traffic load of 600 Erlangs. This approach takes into account only one stratum (DC resources). As shown in Fig. 2(b), this prevents efficient use of optical resources because requests are always sent to the least loaded DC, which may result in sending requests over long distances thus overutilizing optical fiber resources. As a consequence, the DC resource utilization is low as shown in Fig. 2(a) while the cost of such approach is almost twice as high than other approaches shown in Fig. 2(d).

The Nearest DC approach also performs poorly. However, it offers more consistent results albeit at unacceptable levels ($\geq$ 1% blocking percentage) as shown in Fig. 2(c). This approach also takes into account only one stratum (optical resources). By selecting the nearest DC, paths between the source and destination nodes are shorter by approximately 20% compared to other approaches. However, DC resources are used inefficiently as shown in Fig. 2(a).

The Cheapest approach offers better performance. It maintains acceptable blocking percentage and relatively low cost for traffic loads below 700 Erlangs as shown in Fig. 2(c) and Fig. 2(d), respectively. For the traffic load above 850 Erlangs, the cost of this approach is higher than the Nearest. This is due to longer distances between source and destination nodes. These distances are artificially longer because the approach always tries to establish a connection to the cheapest available DC. When the traffic load is high, this approach results in costly decisions because of overloading the cheapest DC at the beginning of simulations.

The two best approaches are Hybrid and Traffic Prediction. While Hybrid is slightly cheaper than the Traffic Prediction approach at higher traffic loads (Fig. 2(d)), Traffic Prediction outperforms Hybrid approach in terms of blocking percentage as shown in Fig. 2(c). Traffic Prediction utilizes DC resources more efficiently than Hybrid (Fig. 2(a)).

In cases of occasional traffic bursts, the Traffic Prediction algorithm ensures the best performance and guarantees proper handling of peaks in traffic load. Moreover, the Traffic Prediction approach preserves the highest residual network capacity. Hence, a larger number of network requests may be served without imposing an additional cost.

We also simulated the US26 network. Due to the space limits, we present only partial results in Fig. 3. The Hybrid and Traffic Prediction approaches significantly outperform the other approaches considered in this paper. The other approaches result in over 5% blocking percentage for low and above 10% for moderate traffic loads, so we skip high traffic loads. The performance of these three approaches in US26 network is inferior to their performance in Euro28 network because the paths between nodes are much longer. Furthermore, DCs are concentrated in the East and the West Coasts in US26 network while DCs in the Euro28 network are more centralized. Hence, because of larger distances, poor decisions more significantly affect the network performance. These poor decisions also result in faster depletion of optical resources and leaving the DCs underutilized. These trends are similar to those observed in the Euro28 network because both Hybrid and Traffic Prediction simultaneously optimize use of resources in two network strata.

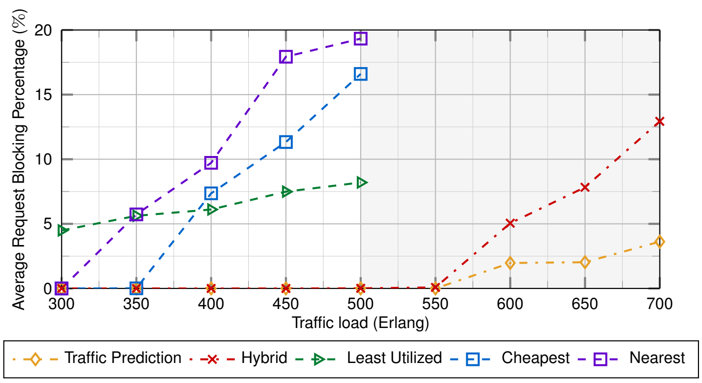

Fig. 3: Performance of the algorithms using the US26 network topology.

VI. Conclusion

We evaluated various approaches to optimize resource allocation using the Cross-Stratum Optimization architecture. While simple approaches that consider only one stratum were unable to deliver required performance, more sophisticated approaches were able to better coordinate resource allocation in both strata thus improving network performance. The cost for using Hybrid approach was slightly lower than the cost of using the Traffic Prediction approach. However, it resulted in high request blocking percentage in case of high traffic loads. The Traffic Prediction approach, albeit delivering a more costly solution, resulted in low request blocking percentage and more efficient utilization network resources.

As future works, we plan to investigate a similar problem with survivability issues included, as well as compare our algorithms with the ones designed for the classical machine scheduling.

Acknowledgment

M. Aibin and K. Walkowiak are supported by the Polish National Science Centre under grant 2016/21/N/ST7/02147, and by the statutory funds of the Department of Systems and Computer Networks, Wroclaw University of Science and Technology.

References

[1] H. Yang, Y. Zhao, J. Zhang, Y. Lee, Y. Lin, and F. Zhang, “Cross stratum optimization architecture for Optical as a Service,” *IEEE Communications Magazine*, vol. 15, no. 8, pp. 130–140, Aug. 2015.

[2] H. Yang, Y. Zhao, J. Zhang, J. Wu, J. Han, Y. Lin, Y. Lee, and Y. Ji, “Multi-stratum resilience with resources integration for software defined data center interconnection based on IP over elastic optical networks,” in *European Conference on Optical Communication, ECOC*, vol. 18, no. 10, 2014, pp. 1735–1738.

[3] X. Yu, M. Tornatore, M. Xia, J. Wang, J. Zhang, Y. Zhao, J. Zhang, and B. Mukherjee, “Migration from Fixed Grid to Flexible Grid in Optical Networks,” *IEEE Communications Magazine*, vol. 53, no. 2, pp. 34–43, 2015.

[4] M. Jinno, H. Takara, B. Kozicki, Y. Tsukishima, Y. Sone, and S. Matsuoka, “Spectrum-efficient and scalable elastic optical path network: Architecture, benefits, and enabling technologies,” Nov. 2009, vol. 47, no. 11, pp. 66–73.

[5] M. Klinkowski and K. Walkowiak, “On the advantages of elastic optical networks for provisioning of cloud computing traffic,” *IEEE Network*, vol. 27, no. 6, pp. 44–51, Nov. 2013.

[6] C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-Defined Networking : A Comprehensive Survey,” in *Proceedings of the IEEE*, vol. 103, no. 1, 2015, pp. 14 – 76.

[7] R. Jain and S. Paul, “Network virtualization and software defined networking for cloud computing: a survey,” *IEEE Communications Magazine*, vol. 51, no. 11, pp. 24–31, Nov. 2013.

[8] L. Lei, “SDN orchestration for dynamic end-to-end control of data center multi-domain optical networking,” *IEEE Communications China*, vol. 12, no. 8, pp. 10–21, Aug. 2013.

[9] L. Kocsis and C. Szepesvári, “Bandit based Monte-Carlo planning,” in *Lecture Notes in Computer Science: Proc. 17th European Conference on Machine Learning (ECML)*, J. Furnkranz, T. Scheffer, and M. Spiliopoulou, Eds. Heidelberg, Germany: Springer, 2006, vol. 4212, pp. 282–293.

[10] L. Kocsis and C. Szepesvári, “Bandit based monte-carlo planning,” in *Proceedings of ECML*, 2006, pp. 282–203.

[11] M. Kearns, Y. Mansour, and A. Y. Ng, “A sparse sampling algorithm for near-optimal planning in large Markov decision processes,” in *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2, 1999, pp. 1324–1331.

[12] T. Cazenave, “Nested Monte-Carlo search,” in *IJCAI International Joint Conference on Artificial Intelligence*, 2009, pp. 456–461.

[13] M. Betancourt, “Nested sampling with constrained Hamiltonian Monte Carlo,” in *AIP Conference Proceedings*, vol. 1305, 2010, pp. 165–172.

[14] C. D. Rosin, “Nested rollout policy adaptation for Monte Carlo tree search,” in *IJCAI International Joint Conference on Artificial Intelligence*, 2011, pp. 649–654.

[15] J. G. March, “Exploration and Exploitation in Organizational Learning,” *Organization Science*, vol. 2, no. 1, pp. 71–88, 1991.

[16] R. Coulom, “Efficient Selectivity and Backup Operators in Monte-Carlo Tree Search,” in *Computers and games*, 2007, vol. 4630, pp. 72–83.

[17] M. P. D. Schadd, M. H. M. Winands, M. J. W. Tak, and J. W. H. M. Uiterwijk, “Single-player Monte-Carlo tree search for SameGame,” *Knowledge-Based Systems*, vol. 34, pp. 3–11, 2012.

[18] Amazon Web Services, “Elastic Compute Cloud (EC2) Cloud Server & Hosting AWS,” 2016. [Online]. Available: https://aws.amazon.com/ec2/