Network routing optimization using Digital Twins

Mohamed Zalat[1], Chris Barber[2], David Krauss[2], Babak Esfandiari[1] and Thomas Kunz[1]

[1]*Carleton University, Ottawa, Ontario, Canada*
[2]*Ciena, Hanover, Maryland, USA*

Abstract
In the context of network Traffic Engineering (TE), the Open Shortest Path First (OSPF) configuration of a communication network is optimized based on the maximum link utilization or the average link utilization of the network. In this paper, we introduce an OSPF Interior Gateway Protocol (IGP) weight optimization technique using a Digital Twin (DT) of the network that optimizes the weights based on Quality of Service (QoS) metrics, specifically the average End-to-End (E2E) traffic delay. Our results show that we can significantly minimize the average traffic delay of a network by optimizing the OSPF configuration using a DT.

Keywords
Traffic Engineering, OSPF, IGP, Optimization, Network Digital Twins, Digital Twins

1. Introduction

1.1. Motivation

The traffic flows in large communication networks are constantly changing and evolving over small time scales [1]. It takes time for experts to manually modify the current configuration of the network to maintain its performance. Failure to do so may result in outages and downtime in the network. To address this, *Traffic Engineering (TE)* is employed to optimize routing configurations of the network for maximum network performance and minimal traffic congestion. Thus, many researchers have proposed TE techniques to quickly identify optimal configurations for the current network state [2, 3, 4, 5, 6] instead of relying on heuristics used by experts.

Rusek *et al.* proposed *Routing by Back-propagation (RBB)* [2]: a technique that uses a *Graph Neural Networks (GNN)* [7] trained to approximate Dijkstra's algorithm to find an Open Shortest Path First (OSPF) configuration that minimizes the maximally utilized link. On the other hand, TE methods proposed in [5] and [6], use local search heuristics to find an OSPF configuration that minimizes the total network utilization and the maximum link utilization respectively. However, optimizing the OSPF configuration based on link utilization does not consider Quality of Service (QoS) metrics and may negatively impact traffic delays.

In recent work, Ferriol-Galmés *et al.* proposed using a GNN model of a network simulator called *TwinNet* that predicts traffic QoS metrics for TE [8]. They use TwinNet to find traffic routes that meet the QoS requirements for each traffic flow, independent of the OSPF configuration.

Companion Proceedings of the 16th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modeling and the 13th Enterprise Design and Engineering Working Conference, November 28 – December 1, 2023, Vienna, Austria
mohamedzalat@sce.carleton.ca (M. Zalat); cbarber@ciena.com (C. Barber); dkrauss@ciena.com (D. Krauss); babak@sce.carleton.ca (B. Esfandiari); tkunz@sce.carleton.ca (T. Kunz)

We take inspiration from their work to introduce an OSPF optimization technique that considers QoS metrics. Using a *Network Digital Twin (NDT)* model, such as TwinNet, one can produce architectures capable of optimizing the OSPF configuration of a network for the QoS metrics of their choice.

1.2. Contributions

To optimize the OSPF configuration of a network based on QoS metrics, we propose using a *NDT*: a Digital Twin (DT) of a communication network that is capable of optimizing the network and reflecting the behavior of the network [9]. Digital Twins (DT) are recently emerging in the field of communication networks and there is a work-in-progress Internet Research Task Force (IRTF) draft on a reference architecture of a NDT [10]. The NDT takes information about the topology (including the OSPF configuration) and traffic flows going through its physical twin (the physical network we are optimizing) and updates the OSPF configuration of its physical twin based on the predicted optimal configuration.

We implement our NDT by combining RouteNet-F [11], a GNN model that predicts QoS metrics for a given network and its traffic flows, with a genetic algorithm designed to minimize traffic delays of an OMNeT++ simulated network (the physical twin) [12]. We show how our method improves the traffic delays of randomly generated network topologies by optimizing their OSPF configuration.

1.3. Structure

In Section 2, we summarize the state-of-the-art TE methods for OSPF optimization. In Section 3, we introduce some background which we base our NDT implementation on. In Section 4, we describe the method we propose to perform OSPF optimization using a NDT. In Section 5, we show how our OSPF optimization method using NDT was able to optimize the simulated network topologies. Lastly, we provide a concluding paragraph with the future work in Section 6.

2. OSPF weight optimization state of the art

OSPF weight optimization is the process of setting the *Interior Gateway Protocol (IGP)* metric of each link in the network to optimize for some metric of the network, such as maximum link utilization, average End-to-End (E2E) loss, etc. The IGP metric of a link in the OSPF protocol, also referred to as the weight of the link, is the cost of taking this link. Traffic flowing to a destination node from a source node is routed through the minimal-cost path.

Current methods in the literature only optimize for bottleneck link utilization or average link utilization [6, 5, 2], or for energy efficiency [13]. One method includes Service-Layer Agreement (SLA) requirements and performance under link failures in addition to link utilization [14]. The problem of finding an OSPF configuration that minimizes the bottleneck link utilization for a given topology and a set of traffic flows/demands is a known *NP-hard* problem [15]. Current methods either use a neighborhood search technique using a set of heuristics, or a Machine Learning (ML) approach. In this section, we cover the state-of-the-art of OSPF optimization for

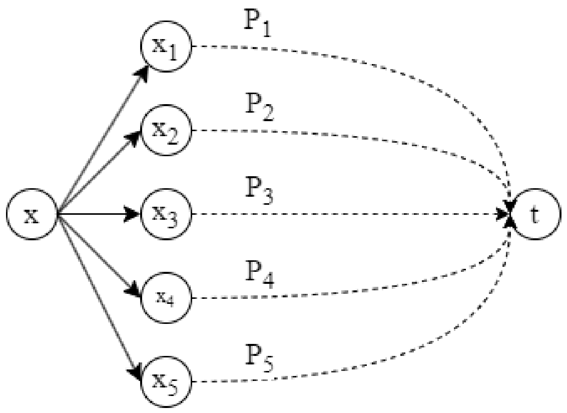


Figure 1: Traffic travelling to node t is split at transit node x to travel though paths P_1 to P_5 to distribute the traffic load. This is done by setting the weight between node x and its 5 neighbors such that the cost of $P_1 = P_2 = P_3 = P_4 = P_5$ [6].

link utilization starting with the first method, *IGP Weight Optimization* [6], a technique that uses local search heuristics to find the optimal OSPF configuration.

2.1. Local Search approach

2.1.1. IGP Weight Optimization

IGP Weight Optimization was proposed by Fortz and Thorup [6]. It takes in the topology of the network, the associated weight and capacity of each link in the network, and the bandwidth demand between each source-destination node pair. It then uses a local search algorithm to find a weight configuration that minimizes the total utilization cost of all links in the network. They define the utilization cost of a link as a function that scales exponentially with the utilization percentage of the link.

The neighbors of the current candidate OSPF configuration of a given network topology are defined as the following:

- Single link weight change.
- Splitting traffic to a destination node evenly across a random number of neighbors at a transit node (refer to Figure 1).

To prevent looping through visited OSPF configurations, they maintain a hash table of all visited OSPF configurations. This algorithm can be terminated at any point during the search to find a better OSPF configuration; however, it is not guaranteed that it will find the global optimum configuration. Rusek *et al.* [2] recently reproduced the results of Fortz and Thorup [6] to compare IGP weight optimization to their ML approach. They show that it can find significantly better OSPF configurations that minimize the maximum link utilization in a short period of time on 4 real-world topologies (Janos-US, GEANT, Nobel-Germany, and COST266).

2.1.2. Dynamic IGP Weight Optimization

Dynamic IGP Weight Optimization was proposed by Brun and Garcia [5]. It uses a very similar local search algorithm as IGP Weight Optimization with some slight modifications to the neighborhood space to find an OSPF configuration that minimizes the maximally utilized link. The neighbors of a candidate OSPF configuration are defined as the following:

- A single link weight change that deviates a traffic flow from the link if the traffic flow has multiple minimum cost paths.
- A single link weight change that introduces an alternative path for a traffic flow (introduces a new minimum cost path).

In addition to those differences, Dynamic IGP Weight Optimization estimates the source-destination demand matrix from a time series of link demands rather than having prior knowledge of those demands. The resulting OSPF configuration recommendation is then based on the worst-case demand scenario out of the set of possible demands inferred from the link demands. However, this makes it much more computationally expensive than IGP Weight Optimization.

2.2. Machine Learning approach

2.2.1. Routing by Back Propagation (RBB)

Routing by Back-propagation (RBB) is the latest OSPF optimization technique proposed by Rusek *et al.* [2] It makes use of a *GNN* model [7] that is trained to approximate the minimum cost path between each source-destination node pair (i.e. Dijkstra's algorithm [16]) for a given topology with a set of link weights. This pre-trained GNN is then used along with the capacity of each link and the traffic demands between each source-destination pair to backpropagate the input IGP weights of each link to minimize the maximally utilized link. This method is much faster than the aforementioned techniques, with the authors reporting an average 25% reduction in the maximally utilized link utilization from the default OSPF configuration after 3 backpropagation steps.

While the methods we listed in this section can find much better OSPF configurations in a short period, none of them consider any variable beyond link utilization for OSPF configuration. Before introducing our NDT for OSPF optimization, we introduce some of the related work used in our NDT implementation in the next section.

3. Background

3.1. RouteNet-Fermi

RouteNet-Fermi, also known as RouteNet-F, is a GNN model of a communication network proposed by the same authors of TwinNet, Ferriol-Galmés *et al.* [11] It takes the topology of the network (which includes link bandwidths, queuing policy of each interface, the number of buffers at each interface and their respective sizes), the traffic flows between each source-destination node (including the average bandwidth used, the packet size distribution and time distribution), and the route each traffic flow takes in the topology as input. The resulting output

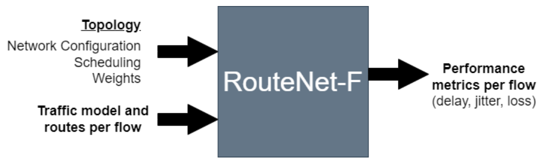


Figure 2: Inputs and outputs of RouteNet-F. [11]

of RouteNet-F is the predicted delay, jitter, and loss of each traffic flow. Their results show that RouteNet-F can generalize and accurately predict QoS metrics on unseen topologies and traffic flows. It serves as a ML based network model that is capable of predicting QoS metrics much faster than tools such as OMNeT++ [12]. For those reasons, we chose to use RouteNet-F as our digital network model in our NDT implementation in Section 4.

In the following section, we introduce *genetic algorithms*, the optimization technique that is coupled with RouteNet-F for our NDT implementation of QoS based OSPF optimization.

3.2. Genetic algorithms

Genetic algorithms are search and optimization algorithms proposed by Holland [17]. They are inspired by the process of natural selection and evolution. They are characterized by the following seven steps:

- **Initialization** where a population of potential solutions for an optimization problem is randomly generated.
- **Evaluation** where each solution is evaluated using a fitness function.
- **Selection** where solutions with a higher score are selected for "reproduction".
- **Recombination** where a pair or pairs of the highest-scoring solutions are randomly combined to create the new "population".
- **Mutation** where solutions in the new population are randomly modified to prevent the algorithm from converging early.
- **Replacement** where new solutions replace the old solutions in the population when creating the next generation.
- **Termination** where the algorithm terminates at a specified stopping criterion.

Genetic algorithms are very simple to implement and are applicable to any fitness function rather than being specialized such as heuristic local search techniques. They are also less likely to be stuck in a local optimum due to the mutation step. However, they come at a computational expense. When using a ML digital model such as RouteNet-F to provide the metrics for our fitness function, we can afford to use genetic algorithms and find a much better solution than the current solution within a short period. We decided to use genetic algorithms for their simplicity and generalization as our goal is to show how we can use a NDT for optimizing OSPF configuration based on QoS metrics. While using a local search technique with heuristics such as those mentioned in Section 2 can improve the time it takes to find a better solution, using a genetic algorithm gives us the flexibility to choose our fitness function. In the following section, we describe the architecture of our NDT for OSPF optimization and the genetic algorithm we use.

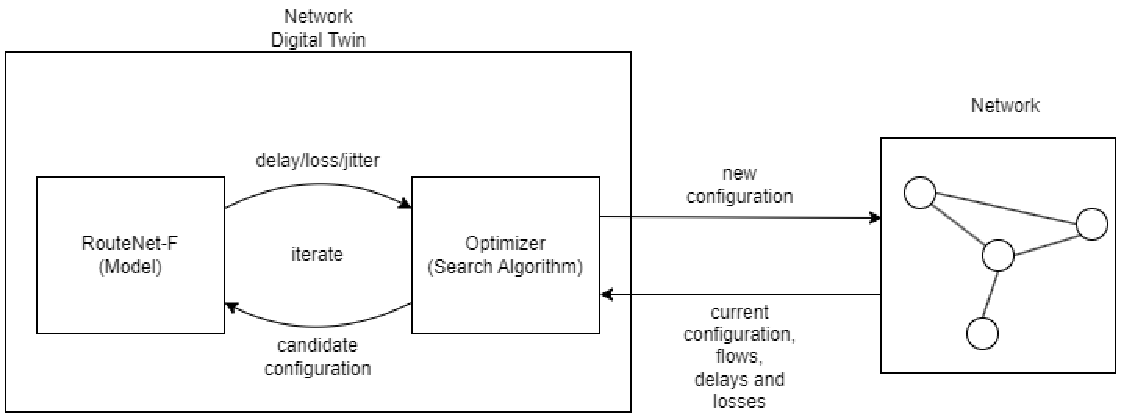


Figure 3: Our proposed OSPF optimization NDT. Here our optimizer is a genetic algorithm for demonstrating our NDT OSPF optimization technique.

4. Methodology

Our proposed NDT architecture for OSPF optimization is shown in Figure 3. We use RouteNet-F as the model to predict the QoS metrics of the current traffic flows going through the network and couple it with a genetic algorithm to find an OSPF configuration that minimizes the average traffic delay in the network. Our genetic algorithm is initialized by mutating the current OSPF configuration to create a population of 2000 solutions in our experiments. Mutations have a 20% chance of occurring on each link in all generations. The solution with the minimum average traffic delay is then recombined with itself to create a new population. If no solution from the new generation has a lower average traffic delay than the previous generation for 4 generations our algorithm terminates. The population size and mutation probability as well as our termination criteria are hyper-parameters of our model and were selected based on our intuition. Algorithm 1 summarizes the implementation of our NDT for OSPF optimization.

To evaluate our implementation, we randomly generated three topologies, a 6-node topology, a 10-node topology, and a 20-node topology with randomly generated Poisson traffic flows ranging from 200 bits per second to 2000 bits per second for each source-destination pair. The link capacity of each link was set to 100kbps and all queuing policies were set to FIFO with a single 32kbits queue. We used the RouteNet-F model trained on the fat-tree $N = 128$ dataset provided in the RouteNet-F paper [11]. For our networks (i.e. the “physical” twin) we used the same OMNeT++ simulation used to train RouteNet-F. The OSPF weights were also randomly initialized to either have a weight of 5 or 1 in our simulated networks. The genetic algorithm was also constrained to this set of weights to minimize its search space. In the following section, we report and discuss the results of our experiments.

5. Results and discussion

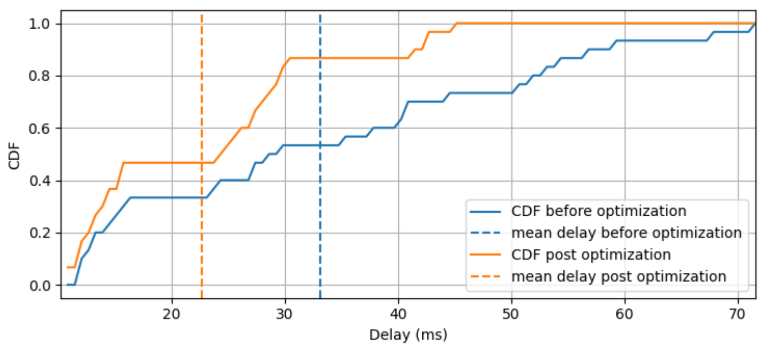
In Figure 4, we plot the Cumulative Distribution Function (CDF) of the traffic flow delays going through each of our simulated topologies before and after optimization. We obtained the delays

Algorithm 1: Digital Twin for Traffic Engineering

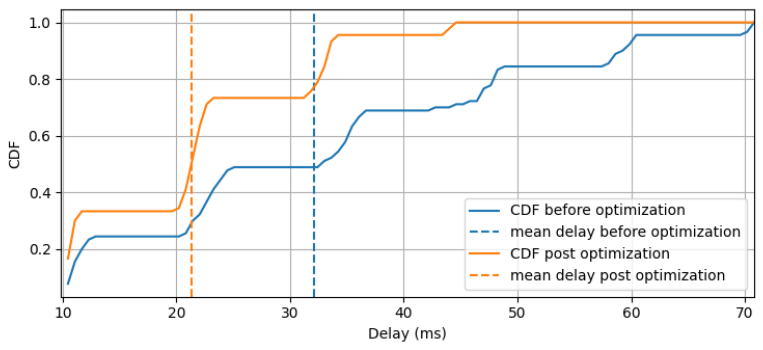
```
Input: $N$, $M$ (where $N$ is the real network, and $M$ is the digital model)
Parameter: $T$ generations without improvement before termination, $s$ size of generation

$topology$ ← retrieve topology and features of $N$
$flows$ ← retrieve estimated distribution of each flow in $N$
$bestConfig \leftarrow topology$
$bestDelay$ ← predicted average flow delay of $N$ using $M$
$generationsWithoutImprovement \leftarrow 0$
while True do
    $candidateConfigs$ ← generate $s$ weight configurations from $topology$ using
      genetic algorithm and recompute routes
    $predictedDelays$ ← predict the average flow delay of $candidateConfigs$ using $M$
    $generationsWithoutImprovement \leftarrow generationsWithoutImprovement + 1$
    if $min(predictedDelays) < bestDelay$ then
        $bestDelay \leftarrow min(predictedDelays)$
        $bestConfig \leftarrow candidateConfigs[predictedDelays.index(bestDelay)]$
        $generationsWithoutImprovement \leftarrow 0$
    end
    if $generationsWithoutImprovement \geq T$ then
        break
    end
end
$actualDelays$ ← Apply $bestConfig$ to $N$ and retrieve average flow delays from $N$
```

from the OMNeT++ simulated topology (i.e. our “network physical twin”) in our experiments. Our results show that our technique successfully reduced the average traffic delay in all our randomly generated networks. Our method shifts the distribution of traffic delays closer to lower delays and consistently reduced the maximum traffic delay in the network in all our topologies. All our NDTs found a solution within 5 minutes on a GTX1080Ti GPU and an Intel i7 7700k CPU. Our promising results indicate that our NDT for OSPF optimization based on QoS metrics is viable and more importantly, allows us to optimize for other QoS metrics beyond delay with ease. We can optimize for other metrics by simply changing the fitness function. For instance, to optimize traffic loss or link utilization, we can change the fitness function to be negatively proportional to the maximum traffic loss or the maximum link utilization in the network without modifying the architecture.

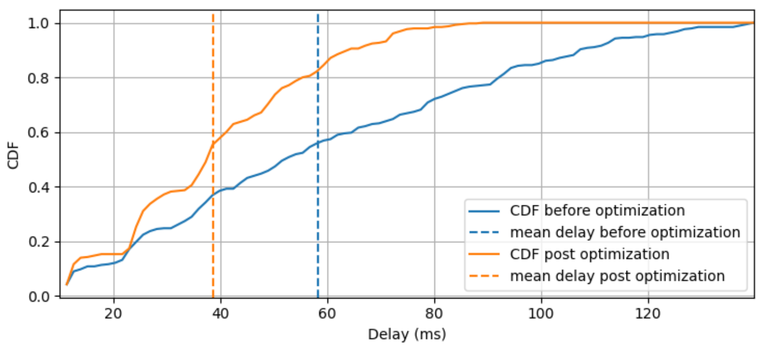
Coupling our NDT architecture for optimizing a network with a smarter optimization algorithm and a desired optimization criteria opens the door for new methods of automating TE and network optimization.



(a) 6 Node topology.



(b) 10 Node topology.



(c) 20 Node topology.

Figure 4: Cumulative Distribution Function (CDF) plots of the traffic delays of each flow in the network before and after optimization using our NDT OSPF optimization technique.

6. Conclusions and future work

In this paper, we showed how one can use our NDT architecture to perform OSPF optimization based on QoS metrics. Our NDT was able to find OSPF configurations that significantly reduced the traffic delays in the twinned networks. However, our genetic algorithm was limited to a set of two weights and may take longer to terminate when the set of weights is large as it increases the search space. A better termination criterion in this case would be the percentage improvement in the last t generations.

Using a smarter optimization algorithm such as a local search heuristic can also improve performance and the speed of finding a solution. However, the genetic algorithm provides a baseline implementation that can be extended further.

Our future work would include comparing our NDT for OSPF optimization to the state-of-the-art OSPF optimization techniques using the average traffic delay, maximum link utilization and the time to arrive at a solution. Moreover, applying this architecture to a real network rather than a simulated network would show how well the NDT for OSPF optimization performs in practice and if changes are required to adjust for a real network.

Acknowledgments

We acknowledge the support of Ciena and the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] Z. Wang, Z. Li, G. Liu, Y. Chen, Q. Wu, G. Cheng, Examination of wan traffic characteristics in a large-scale data center network, in: Proceedings of the 21st ACM Internet Measurement Conference, 2021, pp. 1–14.

[2] K. Rusek, P. Almasan, J. Suárez-Varela, P. Chołda, P. Barlet-Ros, A. Cabellos-Aparicio, Fast traffic engineering by gradient descent with learned differentiable routing, in: 2022 18th International Conference on Network and Service Management (CNSM), IEEE, 2022, pp. 359–363.

[3] S. Gay, P. Schaus, S. Vissicchio, Repetita: Repeatable experiments for performance evaluation of traffic-engineering algorithms, arXiv preprint arXiv:1710.08665 (2017).

[4] R. Hartert, S. Vissicchio, P. Schaus, O. Bonaventure, C. Filsfils, T. Telkamp, P. Francois, A declarative and expressive approach to control forwarding paths in carrier-grade networks, ACM SIGCOMM computer communication review 45 (2015) 15–28.

[5] O. Brun, J.-M. Garcia, Dynamic igp weight optimization in ip networks, in: 2011 First International Symposium on Network Cloud Computing and Applications, 2011, pp. 36–43. doi:10.1109/NCCA.2011.13.

[6] B. Fortz, M. Thorup, Internet traffic engineering by optimizing ospf weights, in: Proceedings IEEE INFOCOM 2000. conference on computer communications. Nineteenth annual joint conference of the IEEE computer and communications societies (Cat. No. 00CH37064), volume 2, IEEE, 2000, pp. 519–528.

[7] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, et al., Relational inductive biases, deep learning, and graph networks, arXiv preprint arXiv:1806.01261 (2018).

[8] M. Ferriol-Galmés, J. Suárez-Varela, J. Paillissé, X. Shi, S. Xiao, X. Cheng, P. Barlet-Ros, A. Cabellos-Aparicio, Building a digital twin for network optimization using graph neural networks, Computer Networks 217 (2022) 109329.

[9] P. Almasan, M. Ferriol-Galmés, J. Paillisse, J. Suárez-Varela, D. Perino, D. López, A. A. P. Perales, P. Harvey, L. Ciavaglia, L. Wong, et al., Digital twin network: Opportunities and challenges, arXiv preprint arXiv:2201.01144 (2022).

[10] C. Zhou, H. Yang, X. Duan, D. Lopez, A. Pastor, Q. Wu, M. Boucadair, C. Jacquenet, Digital Twin Network: Concepts and Reference Architecture, Internet-Draft draft-irtf-nmrg-network-digital-twin-arch-03, Internet Engineering Task Force, 2023. URL: https://datatracker.ietf.org/doc/draft-irtf-nmrg-network-digital-twin-arch/03/, work in Progress.

[11] M. Ferriol-Galmés, J. Paillisse, J. Suárez-Varela, K. Rusek, S. Xiao, X. Shi, X. Cheng, P. Barlet-Ros, A. Cabellos-Aparicio, Routenet-fermi: Network modeling with graph neural networks, IEEE/ACM Transactions on Networking (2023).

[12] A. Varga, Omnet++, Modeling and tools for network simulation (2010) 35–59.

[13] F. Francois, N. Wang, K. Moessner, S. Georgoulas, K. Xu, On igp link weight optimization for joint energy efficiency and load balancing improvement, Computer Communications 50 (2014) 130–141.

[14] A. Nucci, S. Bhattacharyya, N. Taft, C. Diot, Igp link weight assignment for operational tier-1 backbones, IEEE/ACM Transactions on Networking 15 (2007) 789–802.

[15] F. Giroire, S. Pérennes, I. Tahiri, On the hardness of equal shortest path routing, Electronic Notes in Discrete Mathematics 41 (2013) 439–446.

[16] E. W. Dijkstra, A note on two problems in connexion with graphs, Numerische mathematik 1 (1959) 269–271.

[17] J. H. Holland, Genetic algorithms, Scientific American (1992).