Adapting the Virtual Network Topology to Near Future Traffic

F. Morales[1]*, P. Festa[2], M. Ruiz[1], and L. Velasco[1]

(1) Optical Communications Group (GCO), Universitat Politècnica de Catalunya (UPC), Barcelona, Spain

(2) Dipartimento di Matematica e Applicazioni "R. Caccioppoli", Università degli Studi di Napoli Federico II (UNINA), Napoli, Italy

**e-mail:fmorales@ac.upc.edu*

ABSTRACT

The introduction of new services requiring large and dynamic bitrate connectivity can cause changes in the direction of the traffic in metro and even core network segments along the day. This leads to large overprovisioning in statically managed virtual network topologies (VNT), designed to cope with the traffic forecast. To reduce expenses while ensuring the required grade of service, in this paper we propose the VNT reconfiguration approach based on current and near-future traffic matrices (VENTURE) to regularly adapt the topology to both, the current and future traffic volume and direction. The problem is formally stated and a heuristic algorithm is proposed to solve it.

Keywords: virtual network topology reconfiguration, traffic prediction, multilayer networks.

1. INTRODUCTION

Static virtual network topologies (VNT), where large packet-switching nodes (e.g., IP/MPLS routers) are connected through virtual links (*vlinks*) supported by static connections in the optical layer, have been commonly designed to cope with the off-net traffic forecast. However, the introduction of new types of service [1] requiring large bitrate connectivity can cause changes in the direction of the traffic along the day in metro and even core network segments. This, together with the overall traffic increment that operators' networks are needing to deal with year after year entails that static VNT were largely overprovisioned thus, increasing network total cost of ownership (TCO). In view of that, network operators are looking for more efficient architectures able to reduce TCO, while providing the required grade of service. To that end, VNT need to be dynamically adapted to both volumetric and directional traffic variations.

To automate VNT adaptability, traffic needs to be monitored in the packet nodes and the disaggregated traffic volume between every origin-destination (OD) pair of nodes in the network should be available. By applying data analytics techniques to monitoring data, predictive models for the OD traffic can be estimated and used to accurately predict traffic. This prediction is finally used to guide a decision making process responsible to adapt the VNT while in-operation, effectively applying the observe-analyse-act loop [2] in the network and allowing a more efficient use of transponders. In this paper, we develop this idea by presenting the VNT reconfiguration problem based on traffic prediction (VENTURE) that is executed periodically (e.g., hourly) to adapt the VNT based on current and predicted traffic matrices. The VENTURE problem is formally stated and a heuristic algorithm to solve it is devised and evaluated through simulation.

2. VNT DESIGN AND RECONFIGURATION OPTIONS

To alleviate capacity over-provisioning in static VNT, that of vlinks can be adapted over time instead of keeping it constant. Let us assume that the capacity of the existing vlinks can be increased and decreased to follow the traffic variations but no new vlinks can be created or removed, keeping hence the VNT invariant. Traffic can be monitored at IP routers and when the amount of traffic through a vlink reaches some threshold (e.g., 90%) the network controller can increase the capacity of such vlink by setting-up a parallel lightpath between the two IP routers. An example is shown in Fig. 1a. During the last two hours, monitoring traffic between node 6 and every other node in the VNT (labelled as 6->*N*), from node 6 to node 7 (6->7) and from node 6 to every node except node 7 (6->*N*\{7}) are plotted. Figure 1b shows the initial VNT where every vlink is supported by a 100 Gb/s lightpath in the optical layer; the IP/MPLS path for OD 6-7 is also shown. In our example, two threshold violations for vlinks 1-6 and 1-7 are received after the first hour, so the VNT capacity is updated (Fig. 1c). It is worth noting that IP/MPLS path for OD 6->7 is not affected by the VNT reconfiguration. As shown in the example, the threshold-based reconfiguration is able to adapt the VNT capacity to traffic changes, so resources in the optical layer are allocated only when vlinks need to increase their capacity.

Let us go a step further, assuming that instead of monitoring vlink capacity usage, OD traffic is monitored in the routers. Indeed, analysing the plots in Fig. 1a we realize that traffic 6->7 is responsible for the registered traffic increment. In this case, let us assume that new vlinks can be created or removed in addition to increasing the capacity of the existing ones, so the VNT can change. In the VENTURE approach, OD traffic is periodically analysed and the current VNT is reconfigured accordingly, as shown in Fig. 2; OD traffic 6->7 is analysed at t=60 and a maximum value (e.g., 90 Gb/s) is predicted for the next hour. In consequence, a new vlink between nodes 6 and 7 can be created by establishing a lightpath on the optical layer and traffic 6->7 rerouted (Fig. 2b).

Note that this solution reduces two transponders to be installed in router 1 compared to the threshold-based approach. It is clear that this reduction will happen when the amount of traffic is large enough. In particular, when the amount of traffic exceeds the capacity of the installed transponders (e.g., 100 Gb/s) direct vlinks can be created for part of that traffic, while the residual part could be routed through a different IP/MPLS path.

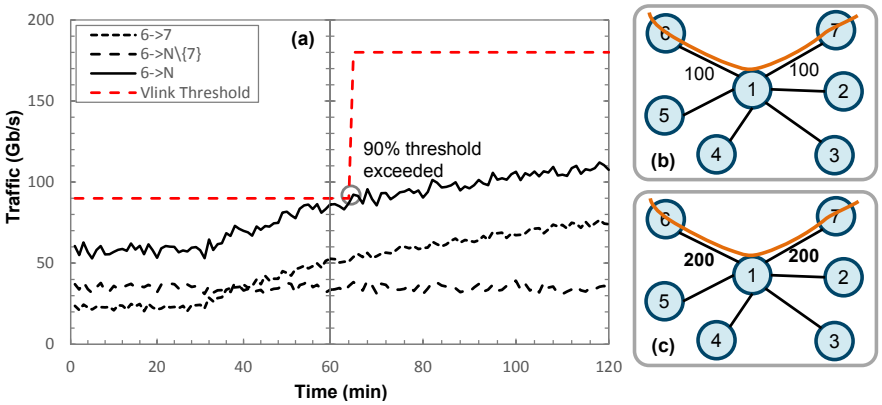

Figure 1. Threshold-based VNT capacity reconfiguration.

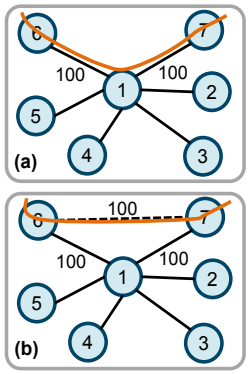

Figure 2. VNT reconfig.

3. THE VENTURE PROBLEM

The VENTURE problem can be formally stated as follows:

Given:

- The current VNT represented by a graph G(N, E'), being N the set of routers and E' $\subseteq$ the set of current vlinks. Set E is the set of all possible vlinks connecting two routers.
- The set P with the transponders available in the routers; every transponder with capacity B.
- The current traffic matrix D.
- The predicted traffic matrix OD. The bitrate b_o of OD pair o must be served following one single path. Only in the case that b_o is enough to fill transponders with an amount over a given boundary usage tbu, the bitrate of pair o can be split into two flows and served through different paths.

Output: The reconfigured VNT $G^*(N, E^*)$, where $E^* \subseteq E$, and the paths for the traffic on G^*.

Objective: Maximize current and predicted served traffic matrices, whilst minimizing the total number of transponders used.

We devise the algorithm in Table 1 to solve the VENTURE problem, consisting of three phases. After deallocating current traffic and releasing used resources (lines 2-5), bitrate b_o of OD pairs is split into two different flows and stored in set Q: flow g_o carries bitrate enough to fill transponders with an amount over tbu and flow l_o carries the remaining bitrate; these flows will be routed through different paths (lines 6-7). Next, the first two phases focus on routing every flow g_o through a direct vlink connecting source and destination nodes (lines 8-9), where set F stores the allocated paths.

Table 1. VENTURE algorithm.

INPUT $G(N,E')$, D, OD, B, tbu
OUTPUT G^*, F

1: $Q \leftarrow \emptyset$, $U \leftarrow \emptyset$
2: **for each** $d \in D$ **do** dealloc(G, d)
3: **for each** $e \in E'$ **do**
4: setCapacity(e, 0)
5: releaseTranspl(e)
6: **for each** $o \in OD$ **do**
7: $Q \leftarrow Q \cup \{<o, g_o, l_o> = \text{splitOD}(o, B, tbu)\}$
8: $<Q, F'> \leftarrow PhaseI(G, Q)$
9: $<G', Q, F''> \leftarrow PhaseII(G, Q)$
10: **for each** $q \in Q$ **do**
11: $U \leftarrow U \cup \{<o, u_o = g_o + l_o>\}$
12: **if** $U=\emptyset$ **then return** $<G', F' \cup F''>$
13: $<G^*, F'''> \leftarrow PhaseIII(G', U, thr)$
14: **if** $F'''=\emptyset$ **then return** INFEASIBLE
15: **return** $<G^*, F' \cup F'' \cup F'''>$

Table 2. Phase I algorithm.

INPUT $G(N,E)$, Q
OUTPUT Q, F

1: $F \leftarrow \emptyset$
2: **for each** $q=<o, g_o, l_o> \in Q$ **do**
3: **if** $e=(s_o, t_o) \notin E$ OR $g_o=0$ **then continue**
4: $n_o \leftarrow \text{ceil}(g_o / B)$
5: $n_s \leftarrow \text{getNumUnusedTranspl}(s_o, P^+)$
6: $n_t \leftarrow \text{getNumUnusedTranspl}(t_o, P^-)$
7: $n \leftarrow \min\{n_o, n_s, n_t\}$
8: allocateTranspl(e, n)
9: $f \leftarrow \text{SP}(G, o, g_o)$
10: **if** $f \neq \emptyset$ **then**
11: allocate(G, f)
12: $F \leftarrow F \cup \{f\}$
13: $g_o \leftarrow g_o - f.b$
14: **return** $<Q, F>$

In the first phase (algorithm in Table 2) flows g_o are routed through existing, direct vlinks in the current VNT, whereas in the second phase the same is done for those not having a direct link, thus creating new ones. After these two phases, the residual bitrate u_o is checked and stored in set U. If all traffic has been already served, the algorithm ends (lines 10-12); otherwise, the third phase eventually routes the unserved bitrate by possibly increasing the capacity of existing vlinks or even creating new ones (line 13, algorithm in Table 3).

In the third phase, the VNT is extended to a full mesh (lines 1-4) and a number of iterations are run targeting at exploring several feasible solutions (line 6). At each iteration, the remaining flows u_o to be allocated are randomly selected introducing a bias in the sorting criterion to process first those flows with higher capacity (lines 12-13). For each selected flow, link metrics are adjusted in order to find the route that minimizes the transponder utilization, allowing to add additional capacity and even new vlinks (lines 15-26); note that a penalty cost is added to the solution cost in case that not all the remaining capacity can be allocated (lines 17-19). Once a solution has been built, a local search procedure is executed (line 27) aiming at finding a local minimum. The best topology and the found paths are eventually returned as final solution (lines 29-32).

Table 3. Phase III algorithm.

INPUT $G(N, E)$, U, thr
OUTPUT $G^*(N, E^*)$, F

1: $G^*(N, E^*) \leftarrow G(N, E)$; $F \leftarrow \emptyset$
2: **for each** $e=(i, j) \notin E \mid i, j \in N, i \neq j$ **do**
3: $E^* \leftarrow E^* \cup \{e\}$
4: setCapacity(e, 0)
5: $bestCost \leftarrow +\infty$
6: **for** $ite = 1 .. maxIter$ **do**
7: $iteUnserved \leftarrow 0$
8: $G_{ite} \leftarrow G$
9: $U_{ite} \leftarrow U$
10: $F_{ite} \leftarrow \emptyset$
11: **for each** $u \in U_{ite}$ **do** $u.order \leftarrow$ rand(0,1) * u_o
12: sort(U_{ite}, $u.order$, DESC)
13: **for each** $u \in U_{ite}$ **do**
14: **if** $u.u_o = 0$ **then continue**
15: updateMetrics(G_{ite}, $u.u_o$)
16: $f \leftarrow$ SP(G_{ite}, $u.o$, $u.u_o$)
17: **if** $f = \emptyset$ **then**
18: $iteCost \leftarrow iteCost + u.u_o * (|P|+1)$
19: **continue**
20: **if** $f.b < u.u_o$ AND canIncreaseCap(f, G_{ite}, $u.u_o$) **then**
21: increaseCap (f, E_{ite}, $u.u_o$)
22: $f.b \leftarrow u.u_o$
23: $F_{ite} \leftarrow F_{ite} \cup \{f\}$
24: allocate(G_{ite}, f)
25: $u.u_o \leftarrow u.u_o - f.b$
26: $iteCost \leftarrow iteCost + u.u_o * (|P|+1)$
27: $<G_{ite}, F_{ite}> \leftarrow$ doLocalSearch(G_{ite}, F_{ite})
28: $iteCost \leftarrow iteCost +$ numUsedTransp(G_{ite})
29: **if** $iteCost < bestCost$ **then**
30: $bestCost \leftarrow iteCost$
31: $<G^*, F> \leftarrow <G_{ite}, F_{ite}>$
32: **return** $<G^*, F>$

4. RESULTS

For evaluation purposes, we ran simulations in an event-driven network simulator implemented in C++. To measure the effect of directional changes in traffic, we injected traffic in the network using the generation framework presented in [4] and assigning different traffic profiles to two subsets of network nodes. We consider an initial full-mesh, 14-node VNT where the capacity of each vlink ranges from 100 to 200 Gb/s and each node is equipped with 26 × 100 Gb/s transponders.

We compared the effect in the unserved traffic and in the number of used transponders under the threshold-based approach (assuming 90% threshold) that runs continuously and under VENTURE that it is triggered on an hourly basis. To predict OD traffic, we use accurate models trained from the generated traffic. The static case where no VNT reconfiguration is performed is also included as a reference.

Figure 3 presents the blocking probability for a range of traffic loads; values for both, the static and the threshold-based approaches are omitted since yield zero blocking probability. In the case of the VENTURE approach Fig. 3a plots the average and maximum hourly blocking along the day. We can observe that the maximum blocking probability remains below 0.25%, being virtually zero on average. Figures 3b-3c analyse the evolution of blocking probability along the day for loads 0.48 and 1, respectively. We observe that small peaks of blocking probability appear related to abrupt changes in the injected traffic and do not last more than two hours, which is the time that VENTURE takes in fully adapting the VNT to traffic changes.

Figure 4a shows the maximum transponder usage in the previous range of loads. Both, the static and the threshold-based approaches show a constant transponder usage for loads lower than 0.5 and increased from that load up. For lower loads, the capacity of vlinks in the fully meshed VNT is 100 Gb/s in both cases and it is increased to 200 Gb/s for high loads under the static approach. The threshold-based approach is able to manage the use of transponders by flexibly using available transponders to increase the vlinks capacity; this way it achieves transponder savings up to 11% with respect to the static approach. Interestingly, transponder usage scales linearly with the load with VENTURE. Compared to the threshold-based approach, VENTURE obtains savings between 8% and 42%. Figures 4b-4c focus on the daily use of transponders for loads 0.48 and 1. Besides

the constant transponder usage in the static approach, let us focus on the different usage of the threshold-based and the VENTURE approaches. In particular, we observe how VENTURE is able to remarkably reduce up to 45% transponder usage at some hours, mainly during night hours. On the other hand, during day hours transponder usage under VENTURE still outperforms that of the threshold-based approach, less adaptive to traffic.

Finally, we evaluated the quality of the VENTURE algorithm by comparing the cost of the obtained solutions (i.e., the transponder utilization) against the optimal solution costs for a set of problem instances corresponding to different hours of the day. To that end, we mathematically modelled the VENTURE problem as a mixed integer linear programming (MILP) problem and solved the instances with a commercial solver. By comparing the optimal solutions against those produced by the VENTURE algorithm, we observed an average optimality gap of 20%, thus opening the possibility of further improving the algorithm.

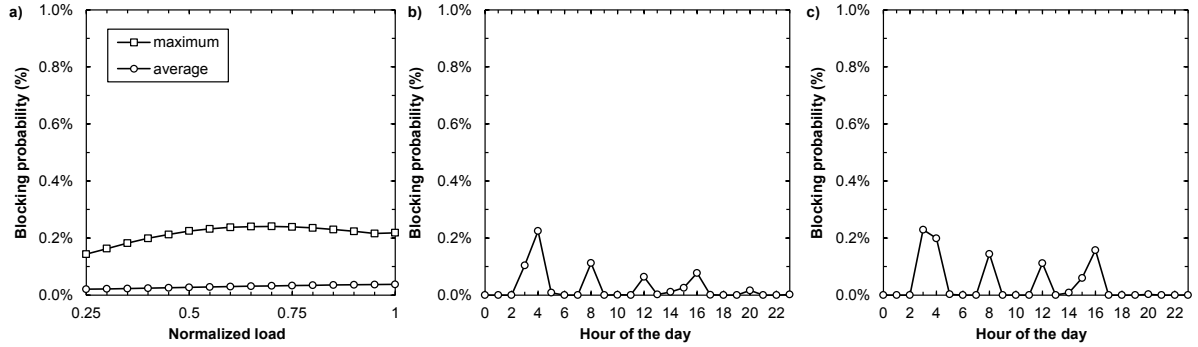

Figure 3. Blocking probability of VENTURE: (a) as a function of the load and (b) along the day.

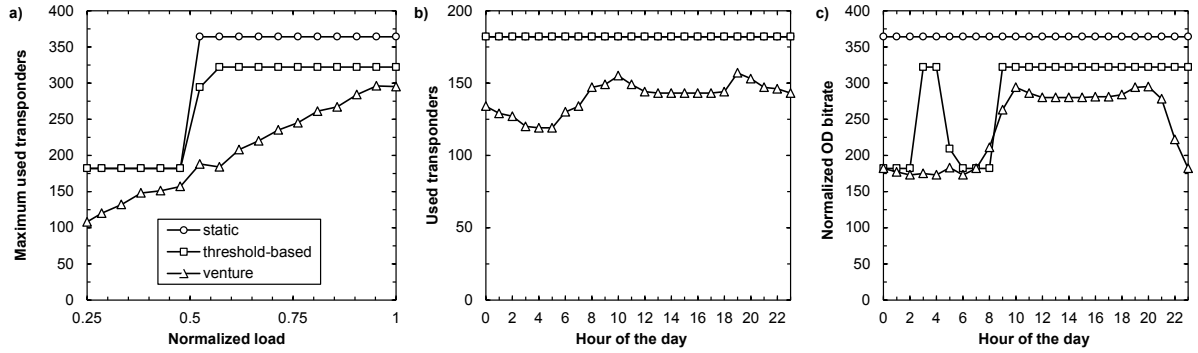

Figure 4. Transponder usage: (a) as a function of the load and (b) along the day.

5. CONCLUSIONS

An approach named as VENTURE has been proposed to adapt the current VNT to current and future traffic. The approach consists of periodically reconfiguring the VNT using short-term predicted OD traffic matrices obtained by applying data analytics to monitoring traffic at IP/MPLS routers. The VENTURE problem was formally stated and a heuristic algorithm proposed to solve it. We compared through simulation the VENTURE approach against a threshold-based VNT reconfiguration, observing savings between 8% and 42% in the number of used transponders. Finally, the quality of the solutions produced by the VENTURE algorithm was analysed, showing an optimality gap of 20%. This opens the possibility of further improving the algorithm.

ACKNOWLEDGEMENTS

This work was partially supported by the EC through the METRO-HAUL project (G.A. nº 761727), from the Spanish MINECO SYNERGY project (TEC2014-59995-R), and from the Catalan Institution for Research and Advanced Studies (ICREA).

REFERENCES

[1] M. Ruiz, M. Germán, L. M. Contreras, and L. Velasco, “Big data-backed video distribution in the telecom cloud,” *Elsevier Computer Communications*, vol. 84, pp. 1-11, 2016.

[2] J. Boyd, “Destruction and Creation,” U.S. Army Command and General Staff College, 1976.

[3] A. Aguado *et al*., “Dynamic virtual network reconfiguration over SDN orchestrated multi-technology optical transport domains,” in *Proc. IEEE ECOC 2015*.

[4] A. P. Vela, A. Via, F. Morales, M. Ruiz, and L. Velasco, “Traffic generation for telecom cloud -based simulation,” in *Proc. IEEE ICTON 2016*.