Virtual Network Reconfiguration in Optical Substrate Networks

F. Gu[1], M. Peng[2], S. Khan[3], A. Rayes[4], N. Ghani[1]
[1]*University of New Mexico*, [2]*Wuhan University*, [3]*North Dakota State University*, [4]*Cisco Systems Inc*

Abstract: This paper studies virtual network service provisioning over substrate optical networks and presents a novel reconfiguration scheme to improve efficiency. Simulation results show lower blocking and increased revenues as compared to existing strategies.

OCIS codes: (060.4250) Networks; (060.4258) Networks, topology

I. Introduction

The continued evolution of user applications, particularly cloud-computing, is driving the need to coordinate geographically-dispersed resources. As a result, network virtualization services have become a key requirement as they allow operators to provision dedicated *virtual networks* (VN) over base optical networking substrates and data-centers. Now a typical user VN request consists of an arbitrary set of VN nodes interconnected by an arbitrary set of VN links, i.e., each VN node requests a certain amount of resources (storage, computing) and each VN link requests a certain amount of bandwidth connectivity between its endpoint VN nodes. Hence operators must implement appropriate VN mapping algorithms to assign VN nodes to distinct *physical* substrate nodes and VN links to underlying (optical) connections. As this mapping problem is NP-hard [2], most existing schemes have proposed heuristics-based strategies to achieve various objectives, e.g., such as revenue maximization or cost reduction [1],[2].

Nevertheless, most VN mapping schemes give high resource fragmentation at the network substrate layer when user requests arrive/depart in a dynamic manner. Hence further VN *reconfiguration* strategies have been proposed to improve resource efficiency. For example, the scheme in [1] periodically migrates selective VNs based upon overloaded substrate fiber links. Meanwhile [2] periodically checks and re-computes VN links for active VN requests with longer residual lifetimes. In general, these periodic strategies are classified as *proactive* and entail relatively high computational overheads and migration costs. Hence some others have also proposed *reactive* VN reconfiguration after setup failures. For example [3] presents a *virtual network reconfiguration* (VNR) scheme that migrates a single node (and its associated VN links) in order to control migration costs, i.e., as opposed to migrating a whole VN topology as in [1]. Simulation findings show good blocking and migration cost reduction with this reactive approach, i.e., versus some proactive schemes. However the VNR scheme implements a greedy approach as it only migrates a minimum number of VN nodes to accept the rejected VN request. As a result this solution may yield increased congestion at certain bottleneck links, leading to higher blocking/reduced revenues.

To address these concerns, this paper presents an improved VN reconfiguration solution for optical networks. The scheme uses a reactive approach to perform multiple (batch) reconfigurations and thereby achieves further improvements in blocking and migration cost. Overall, the paper is organized as follows. Section II details the VN reconfiguration problem followed by the proposed reconfiguration solution. Performance results are then presented in Section III along with conclusions and future work directions.

II. Network Model and VN Reconfiguration

A novel VN reconfiguration scheme is now presented. Consider the requisite notation first. The base optical network substrate is modeled as an undirected graph, $\boldsymbol{G}_s=(\boldsymbol{V}_s,\boldsymbol{E}_s)$, where $\boldsymbol{V}_s$ is the set of substrate nodes and $\boldsymbol{E}_s$ is the set of substrate links. Here each substrate node $v_s \in \boldsymbol{V}_s$ has varying amounts of computing and storage resources, generically given by $R(v_s)$. Furthermore, each substrate fiber link $e_s \in \boldsymbol{E}_s$ also has a variable amount of available capacity given by $B(e_s)$. Meanwhile a client VN request is given by an undirected graph, $\boldsymbol{G}_v=(\boldsymbol{V}_v,\boldsymbol{E}_v)$, where $\boldsymbol{V}_v$ is the set of VN nodes and $\boldsymbol{E}_v$ is the set of VN links. Here each VN node $v_v \in \boldsymbol{V}_v$ requires $r(v_v)$ in nodal resources and each VN link $e_v \in \boldsymbol{E}_v$ requires $b(e_v)$ in bandwidth capacity. Furthermore, a VN mapping requires that a VN node v_v be assigned to a specific substrate node v_s, i.e., denoted as $<v_v,v_s>$. Likewise, each VN link e_v also has two end-points, denoted by the pair (v_v, v_v'). Using the above notation, a sample 8-node optical substrate network is shown in Fig. 1a, where the values over the links (nodes) denote the available bandwidth (nodal resources) levels. A sample 4-node VN request is also shown in Fig. 1b, and this request can be mapped (embedded) over the physical substrate topology.

The proposed VN reconfiguration scheme uses a reactive approach and is termed as the *multiple VN node reconfiguration* (MVNNR) solution. This algorithm is invoked after failure of a regular VN setup attempt and tries to migrate a subset of VN nodes (from the set of existing mapped VN requests) to relieve congestion on overloaded optical substrate links. The detailed psuedocode listing for this algorithm is shown in Fig. 2 and starts by grouping all congested substrate optical links into a *congestion set*, $\boldsymbol{S}$, based

978-1-55752-962-6/13/$31.00 ©2013 Optical Society of America

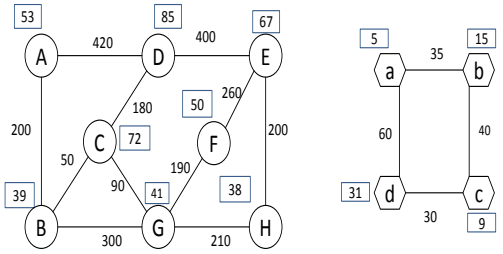

Figure 1: a) substrate network, b) sample VN request

upon their relative bandwidth loading levels, i.e., $\boldsymbol{S}=\{e_s \in \boldsymbol{E}_s \mid B(e_s) \leq \alpha B_c\}$, where B_c is the full link capacity and α is a relative usage threshold, $0 \leq \alpha \leq 1$. Using this set, a ranking metric is also defined for each VN node q as follows:

$$k = A \times T \qquad \text{Eq. (1)}$$

where A is the number of congested links in $\boldsymbol{S}$ used by VN links attached to q, and T is the residual lifetime of the VN request with node q (i.e., since studies have shown the benefit of incorporating lifetimes [2]). This cost k is then used to sort all VN nodes (in all active requests) in decreasing order to generate a *candidate migration list*, $\boldsymbol{Q}$ (Fig. 2). The MVNNR scheme then loops through the first R VN nodes in $\boldsymbol{Q}$ and attempts to migrate each (along with its set of adjacent VN links) in sequence. Namely, a further candidate set of *substrate* nodes, $\boldsymbol{L}$, is generated for each VN node $q_i \in \boldsymbol{Q}$ based upon availability, i.e., if the substrate node is not already mapped to another VN node in the same request and there are sufficient nodal resources (lines 9-12, Fig. 2). The algorithm then uses this candidate set to select a new substrate mapping for VN node q_i. Consider the details here.

```
1. Construct congested link set S and VN node migration list
   Q containing all VN nodes of all active VN requests
2. For each q∈Q, compute migration metric k using Eq. (1)
3. Sort VN nodes in Q in decreasing order using k
4. for i = 1 to R
5.     Select i-th VN node q_i ∈ Q (q_i belongs to request G_v)
6.     n=record of current node/link mapping for <q_i, v_s>, i.e.,
       where v_s is allocated to q_i
7.     Release resource for record n in G_s
8.     Set candidate substrate node list L to empty
9.     for each v_s∈V_s
10.        if (no other VN nodes in G_v mapped to v_s)
11.            if (R(v_s)≥r(q_i))
12.                Insert v_s to L
13.    m=call MVN (q_i,G_v,L)
14.    if (m!=NULL)
15.        Migrate q_i according record m, reserve resource for
           record m in G_s, return SUCCESS
16.    else
17.        Reserve resource for record n in G_s, return FAIL
```

Figure 2: MVNNR algorithm

```
1.  Given VN node q_i in G_v and substrate node list L
2.  Initialize s_min=+∞, m={}
3.  for (each v_s∈L)
4.      Create substrate graph G_s copy, denoted by G_t
5.      l=SUCCESS
6.      for each v_v is adjacent to q_i in G_v
7.          result=compute minimum cost substrate fiber
            path p for virtual link (v_v, q_i) in G_t
8.          if(result=FAIL)
9.              l=FAIL
10.             break
11.         else
12.             reserve bandwidth for p in G_t
13.     if (l=SUCCESS)
14.         s=compute stress for <q_i,v_s> using Eq. (2)
15.         if (s <s_min)
16.             Update s_min = s
17.             m=record of node/link mapping for <q_i, v_s>
18. return m
```

Figure 3: MVN sub-algorithm

The psuedocode for substrate node selection is shown in Fig. 3 and is termed as the *migrate virtual node* (MVN) sub-algorithm, i.e., called from the main MVNNR algorithm (line 13, Fig. 2). This selection process basically looks at each substrate node $v_s \in \boldsymbol{L}$ and tries to map all the attached VN links, i.e., connections to neighboring VN nodes for the given node $q_i \in \boldsymbol{Q}$. Note that this computation is done over a temporary working copy, $\boldsymbol{G}_t$, of the main substrate graph, $\boldsymbol{G}_s$. Now if substrate connection paths can be computed for all VN links here, then the associated substrate node v_s is deemed a valid mapping node. However in order to minimize congestion on substrate optical links, the algorithm also computes a stress metric, s, for all valid mapping nodes and selects the substrate node with the lowest stress from all nodes in $\boldsymbol{L}$, i.e., lines 13-17, Fig. 3. Specifically, this stress value is defined in an analogous manner to [3] as:

$$s = \frac{A}{B_a + \varepsilon} \qquad \text{Eq. (2)}$$

where B_a is the average residual bandwidth of substrate paths allocated for all attached VN links for q_i and ε is a small value to avoid division errors. Carefully note that the shortest-path computation step (i.e., line 7, Fig. 3) can either use fixed (operator-specified) or dynamic link weights in the substrate network graph $\boldsymbol{G}_s$. In particular, the latter case can be used to implement load-balancing operation over the substrate network by assigning link weights as inversely-proportional to available capacity levels. Also note that special data structures are needed to record the mapping information for a VN node and its attached VN links, i.e., as per line 6 in MVNNR (Fig. 2) and line 17 in MVN (Fig. 3).

III. Performance Evaluation

The proposed VN reconfiguration scheme is tested using custom-built models in *OPNET Modeler*^{TM}. All tests are done for a 24-node optical network topology in which substrate nodes have 100 units of generic resource capacity (computing, storage) and substrate optical links have 10,000 units of bandwidth. Meanwhile, VN requests are generated by composing randomized graph topologies with 4-7 nodes each and an average node degree of 2.6. The requested VN node capacities are also uniformly-distributed between 1-10 units and VN link capacities between 50-1,000 units. Furthermore, VN request holding and inter-arrival times are exponentially distributed with means μ=600 sec and λ, respectively (and λ varied according to input load). Here all tests are done for 100,000 VN requests and fixed unity weights are used for all links in the shortest-path computation phase, i.e., to minimize hop count/resource utilization. Now the actual input loads are measured by defining a *modified* Erlang metric that also takes into account VN request sizes, i.e., specified as a product of the average number of VN links and μ/λ. Finally, the *non-survivable virtual infrastructure mapping* (NSVIM) scheme in [4] is used as the base VN mapping algorithm, and the VNR scheme from [3] is also tested for comparison purposes. Specifically, this latter algorithm uses a *single-stage* approach which maps a VN node to a substrate node along with its attached VN links in the same step/iteration.

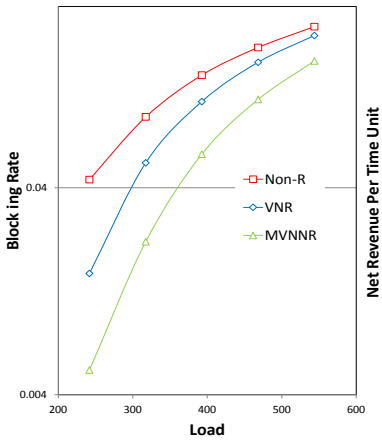

Figure 4: Blocking Rate

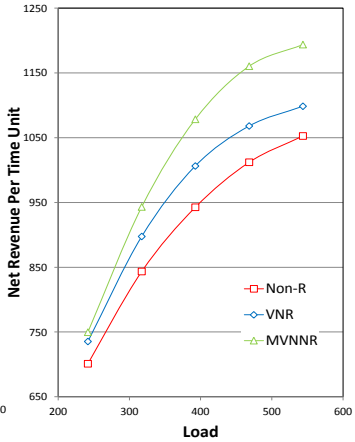

Figure 5: Long Time Net Revenue

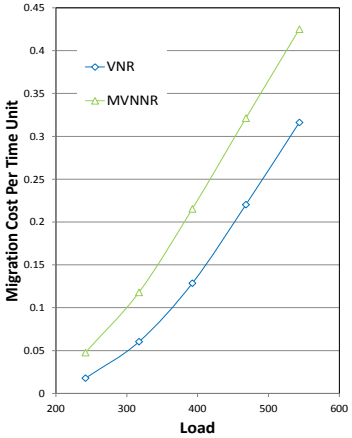

Figure 6: Long Time Reconfiguration Cost

Initial tests are done to measure VN request blocking rates, as shown in Fig. 4 (where the baseline "non-reconfiguration" NSVIM scheme is labeled as "*Non-R*"). These overall findings clearly show much-improved performance with the reconfiguration-based schemes, e.g., up to 88% lower request rejection rates at low-medium loads. More importantly, the proposed MVNNR scheme also gives 66% lower blocking than the VNR scheme, validating the benefits of migrating multiple VN nodes. Next, long term net revenues are plotted in Fig. 5. These findings clearly indicate much higher operator income with the reconfiguration-based schemes, i.e., 13% more at medium-to-high loads. In addition, the MVNNR scheme also gives higher net revenues than the VNR scheme (by almost 10%) due to its lower blocking rates. Finally, Fig. 6 plots the long-term VN reconfiguration costs for the two reconfiguration schemes. Akin to [3], this value is measured as the total cost of migrating all VN nodes and their attached links, i.e., which is given by the weighted sum of the number of migrated VN nodes and VN links. As expected, the proposed MVNNR solution gives higher costs, particularly at heavier loads (up to 34%). Nevertheless, this increase is notably offset by the increase in revenues, i.e., Fig. 5.

This paper studies VN reconfiguration in optical networks and presents a novel reactive algorithm to migrate multiple VN nodes to minimize substrate link congestion. Overall, simulation results show much-improved blocking rates as well as higher net revenues with the proposed scheme (as compared with existing schemes). Future efforts will look at extending this approach to for survivable VN provisioning.

References

[1] Y. Zhu, M. Ammar, "Algorithms for Assigning Substrate Network Resources to Virtual Network Components," *IEEE INFOCOM 2006*, Barcelona, Spain, April 2006.

[2] M. Yu, Y. Yi, J. Rexford, M. Chiang, "Rethinking Virtual Network Embedding: Substrate Support for Path Splitting and Migration," *ACM SIGCOMM Computer Communication Review*, Vol. 38, No. 2, April 2008, pp. 17-29.

[3] I. Fajjari, N. Aitsaadi, G. Pujolle, H. Zimmermann, "VNR Algorithm: A Greedy Approach for Virtual Networks Reconfigurations," *IEEE GLOBECOM 2011*, Houston, TX, December 2011.

[4] H. Yu, *et al*, "A Cost Efficient Design of Virtual Infrastructures with Joint Node and Link Mapping," *Journal of Network and Systems Management*, Vol. 20, No. 1, 2012, pp. 97-115.