Flexible Advance Reservation Models for Virtual Network Scheduling

H. Bai[1], F. Gu[2], K. Shaban[3],J. Crichigno[4], S. Khan[5], N. Ghani[1]
[1]Univ. of South Florida, [2]VMware Inc, [3]Qatar Univ., [4]Northern New Mexico College, [5]North Dakota State Univ.

***Abstract*—Advance reservation services allows users to pre-reserve network resources at future instants in time. These offerings are already being used by a wide range of applications in scientific/grid computing, datacenter backup, and event broadcasting. Now most advance reservation algorithms are designed to schedule point-to-point connection requests. However, as new cloud-based services gain traction, there is a further need to schedule broader virtual network (infrastructure services) demands. These latter types are much more complex and comprise of an arbitrary number of virtual nodes mapped onto physical nodes and interconnected via a set of virtual link connections. This paper addresses this critical area and develops/analyzes several virtual network scheduling schemes, including partial provisioning strategies to improve network revenues.**

***Index Terms*—Virtual networks, scheduling, advance reservation, cloud scheduling**

I. Introduction

The *advance reservation* (AR) services concept was first introduced for optical networks in [1]. The main objective here was to guarantee network resources for customers at future points in time by using time scheduling techniques, i.e., specific start and stop time windows. Over time, AR services have gained increasing favor for a range of applications in scientific computing, datacenter backup, and broadcasting. Overall, these service models contrast with more traditional *immediate reservation* (IR) designs in which user connections are provisioned (in an on-demand or batch manner) using the currently-available (i.e., immediate) set of resources.

Now a truly wide range of AR schemes have been developed, using both heuristic and optimization-based strategies, see [2]. However, due to high computation complexities, most network scheduling problems have only looked at point-to-point connections. Indeed, few (if any) have looked at reserving more complex "virtual network" type demands. For example, [3] introduces the *virtual overlay network scheduling* (VONS) problem to schedule multi-point topology overlays between fixed sites. However, as cloud services continue to evolve, there is a need to expand AR service models to include more generalized *virtual network* (VN) requests. Specifically, this involves mapping virtual nodes and scheduling their virtual links over physical infrastructures at future instants in time.

In addition, further service modifications can also be considered. For example, in many instances customers may accept partial VN service setups if their full requests cannot be provisioned. Namely, simple binary (yes/no) scheduling schemes may lead to lower revenues for operators. Motivated by these scenarios, this paper presents several advance reservation models for VN scheduling, including two partial provisioning strategies, i.e., *priority-based reservation* (PBR) and *capacity-based reservation* (CBR). Overall results show notable gains in terms of blocking reduction and revenues.

This paper is organized as follows. Section II presents a brief overview of existing work in advance reservation. Section III then introduces the required notation and details the proposed VN scheduling schemes. Finally Section IV presents some simulation results, and conclusions and future work directions are also highlightened in Section V.

II. Related Work

AR scheduling involves reserving network resources at future intervals in time. As opposed to many IR-type services, here it is assumed that the duration of user requests/transfers is finite. Now a wide variety of AR (connection) service models have been studied, including fixed start/stop times, variable start/stop times, etc [4]. For the most part, point-to-point AR scheduling problems have been shown to be *NP*-complete. Hence related AR scheduling solutions have used both optimization and heuristic strategies. In particular, the former techniques assume time-slotted arrivals/departures and pursue objectives such as minimizing resource utilization, maximizing the number of accepted requests, etc [5],[6]. However, optimization-based strategies pose notable scalability concerns due to increased variable counts (from the discretized timeline dimension).

Now unlike IR demands, admitted AR reservations remain *inactive* until their future scheduled time in-

tervals. Hence these services can be *re-routed* before activation without disrupting users. As a result several AR rerouting studies have also been conducted [7]-[9]. Now most schemes here use graph-based heuristics and try to achieve a tradeoff between the number of re-routing attempts and blocking reduction. For example, [9] develops an ILP model to re-optimize (re-route) scheduled demands to accomodate a new request. However, since re-optimization complexity can grow quickly, this solution is only feasible for very small network sizes.

Overall, most AR studies have only looked at point-to-point connection demands. However, there is a growing need to schedule more advanced request types. Along these lines, [3] formulates the *virtual overlay network scheduling* (VONS) problem to reserve overlay request topologies with fixed endpoint nodes, i.e., batches of virtual link connections. However, the VONS model does not suffice for cloud-based applications which require more flexibility in mapping (virtual) nodes to network (datacenter) sites with adequate resources, i.e., VN mapping problem. Hence there is a further need to develop more expansive VN scheduling models and algorithms.

III. Flexible Advance Reservation Models

In the light of above, several VN scheduling algorithms are presented. Owing to the complexity of this area, only heuristics-based schemes are considered. A base scheduling algorithm is introduced, followed by more specific priority- and capacity-based variants.

A. Problem Formulation & Baseline Heuristic

A baseline heuristic for VN scheduling is detailed here. Consider the requisite notation. The physical substrate network is specified as an undirected graph $G_s = (V_s, E_s)$, where $V_s = \{v_s^1, v_s^2, ..., v_s^{|V_s|}\}$ is the set of substrate nodes and $E_s = \{e_s | e_s = (v_s^i, v_s^j) : v_s^i, v_s^j \in V_s\}$ is the set of substrate links. Here each substrate node, $v_s \in V_s$, has a fixed amount of computing and storage resources, C, and each substrate link, $e_s \in E_s$, has a fixed bandwidth capacity, B (note that assigning equivalent capacities here does not limit generality). In order to provision VN node/link requests, time-varying capacity levels are also defined for nodes and links, i.e., $rem_v(t)$ and $rem_e(t)$, respectively. Meanwhile a future VN request is given by the 5-tuple, $r^n = (G_v^n, c^n, b^n, t_s^n, t_e^n)$, where n is the request index, $G_v^n = (V_v^n, E_v^n)$ is an undirected virtual network graph containing the set of VN nodes $v_v^n \in V_v^n$ and virtual links $e_v^n \in E_v^n$, c^n is the requested amount of node resources at each VN node ($c^n < C$), b^n is the requested bandwidth for each VN link ($b^n < B$), t_s^n is the start time, and t_e^n is the end time. Hence to schedule r^n, operators must select physical nodes on which to map the virtual nodes and also route connections for the VN links (in the desired interval). Here a VN node-to-substrate mapping is denoted by $< v_v, v_s >$, i.e., VN node v_v mapped to node v_s.

Algorithm 1 Baseline VN Scheduling Algorithm

1: Given incoming request $r^i = (G_v^i, c^i, b^i, t_s^i, t_e^i)$, generate temporary graph copy $G_s'(V_s', E_s') = G(V, E)$
2: Remove non-feasible nodes and links in $G_s'(V_s', E_s')$, i.e., $rem_v(t) < c^n$ and $rem_e(t) < b^n$ in $[t_s^i, t_e^i]$;
/ Loop and map all virtual nodes in the request*/*
3: Sort the virtual nodes based upon node degree
4: **for** $i = 1$ to $|V_v^i|$
5: Randomly pick substrate node from V_s' with sufficient available node resources
6: **if** Failed
7: VN request r^i failed
8: Discard $G_s'(V_s', E_s')$, temporary node mapping array
9: Exit loop
10: **else**
11: Remove substrate node from V_s'
12: Save node mapping $< v_v^i, v_s >$ in temporary node mapping
13: **end if**
14: **end for**
/ Loop and map all virtual nodes in the request*/*
15: **if** All virtual node mappings successful
16: Assign unity weight to links in $G_s'(V_s', E_s')$
17: Run Dijkstra's shortest-path for each virtual link between mapped substrate nodes
18: **else**
19: VN request r^i failed
20: Drop r^i, discard $G_s'(V_s', E_s')$ and node mapping array
21: **end if**
22: **if** All VN link connections routed
23: Setup successful
24: Reserve mapped node resources from node mapping array onto $G_s(V_s, E_s)$
25: Reserve routed link resources onto $G_s(V_s, E_s)$
26: **end if**

The baseline heuristic is shown in Algorithm 1 and uses a two-stage mapping approach. First, the VN nodes are sorted according to node degrees and scheduled sequentially (steps 3-14, Algorithm 1). Next, the virtual links between the mapped nodes are routed/scheduled at future intervals (steps 15-21, Algorithm 1). The scheme only returns success if all VN nodes and VN links are scheduled. Carefully note that a temporary copy of the network graph, $G_s'(V_s', E_s')$ is needed to perform these computations. This graph first removes any non-feasible physical nodes or links which cannot accomodate a virtual node or link, i.e., $rem_v(t) < c^n$ and $rem_e(t) < b^n$.

B. Advance Reservation Models

As mentioned earlier, many users will accept some level of flexibility in meeting their demands. For example, it may be amenable to provision a part of the VN request under high-load or failure conditions. Along these lines, two different *partial* VN scheduling policies are proposed here. Namely, *priority-based reservation* (PBR) assigns different priorities to VN nodes and links. Meanwhile, *capacity-based reservation* (CBR) adjusts request durations to meet revenue expectations. Consider the details.

1) ***Priority-Based Reservation (PBR)***: This scheme assumes two different priority levels for VN nodes/links, i.e., high/low. These delineations can be specified by operators or clients. Hence the algorithm (Algorithm 2) first tries to schedule all higher priority nodes/links. If this is successful, further attempts are made to schedule as many of the lower priority VN nodes/links. Specifically, candidate physical nodes for each unmapped lower priority VN node are selected based upon their costs (route lengths) to each mapped VN node, and this is done in decreasing sequence of VN node degree. The VN link connections between this selected node and the already-mapped nodes are then routed/scheduled. If the whole request can be provisioned, it is marked as a complete success, otherwise it is marked as a partial success.
2) ***Capacity-Based Reservation (CBR)***: This scheme implements a compromise between the assigned resources and request durations. Namely, this algorithm is triggered only when a request cannot be reserved via Algorithm 1. Specifically, all VN node/link resources requested in r^i are adjusted by a fraction, σ, $0 < \sigma < 1$. To compensate for this reduction, the request duration is then extended by $1/\sigma$. The CBR scheme is shown in Algorithm 3. Note that this reservation model may be very useful for transfer services with variable durations.

Algorithm 2 Priority-Based Reservation

```
Given incoming request r^i = (G_v^i, c^i, b^i, t_s^i, t_e^i), generate temporary graph copy G'_s(V'_s, E'_s) = G(V, E)
Remove non-feasible nodes and links in G'_s(V'_s, E'_s), i.e., rem_v(t) < c^n and rem_e(t) < b^n in [t_s^i, t_e^i];
Run Algorithm 1 to schedule higher-priority portion of the request r^i
if Failed
    VN request r^i failed
    Discard G'_s(V'_s, E'_s), temporary node mapping
else
    Setup successful
    Sort the virtual nodes based upon node degree
    for Each unmapped VN node
        Compute candidate physical substrate nodes
        Compute cost (route length) from candidate nodes to mapped VN nodes
        Pick the candidate node with the minimum cost
        Run Dijkstra's shortest-path for the link between picked node and mapped substrate nodes
        if All VN link connections routed
            Reserve mapped node resources from node mapping array onto G_s(V_s, E_s)
            Reserve routed link resources onto G_s(V_s, E_s)
        end if
    end for
end if
```

IV. Evaluation Metrics

Some of the key metrics used to study VN performance are now presented. Network operators provisioning VN services will likely want to achieve high resource efficiency over their underlying substrate networks. In addition, associated revenue generation (cost reduction) concerns will also be very important. Hence two key metrics are introduced here. Foremost, a modified revenue is defined for provisioning a VN request:

$$Revenue(G_v^n) = (t_e^n - t_s^n) * (\sum_{e_v^n \in E_v^n} b^n / \mathbb{B} + \rho \sum_{v_v^n \in V_v^n} c^n / \mathbb{C}) \quad (1)$$

where ρ is the fraction of node resource revenue, and $\mathbb{B}$ and $\mathbb{C}$ are two large numbers to normalize node/link resouces. Unlike [10], here revenue is dependent upon request durations as well, i.e., a request with longer duration but the same amount of node/link resources wil have higher revenue. Similarly, the cost of accepting a VN request is defined as:

$$Cost(G_v^n) = (t_e^n - t_s^n) * (\sum_{e_s \in E_s} \mathcal{F}_{e_s}^{G_v^n} / \mathbb{B} + \pi \sum_{v_s \in V_s} \mathcal{N}_{v_s}^{G_v^n} / \mathbb{C}) \quad (2)$$

where π is the fraction of node resource cost, $\mathcal{F}_{e_s}^{G_v^n}$ is the total amount of bandwidth allocated on substrate link e_s for mapping the VN, and $\mathcal{N}_{v_s}^{G_v^n}$ is the total amount of node resources allocated on the substrate node v_s for mapping the requested VN.

Note that other metrics can also be considered here. For example, VN request blocking ratios can provide very intuitive measurements, and operators may want to minimize these values to improve performance. Additionaly, average VN link connection lengths can be used to gauge network resource usage, and revenue-cost ratios can be used to gauge efficiency.

Algorithm 3 Capacity-Based Reservation

Given incoming request $r^i = (G_v^i, c^i, b^i, t_s^i, t_e^i)$, generate temporary graph copy $G_s'(V_s', E_s') = G(V, E)$
Remove non-feasible nodes and links in $G_s'(V_s', E_s')$, i.e., $rem_v(t) < c^n$ and $rem_e(t) < b^n$ in $[t_s^i, t_e^i]$;
Run Algorithm 1 to schedule request r^i
if Failed
    Scale request resource and duration by σ and $1/\sigma$, $r_\sigma' = (G_v^i, \sigma c^i, \sigma b^i, t_s^i, t_s^i + 1/\sigma(t_e^i - t_s^i))$
    Generate temporary copy of network $G_s''(V_s', E_s')$ where $rem_v(t) < \sigma c^n$ and $rem_e(t) < \sigma b^n$ in $[t_s^i, t_s^i + 1/\sigma(t_e^i - t_s^i)]$
    Run Algorithm 1 to schedule request r_σ'
    if Failed
        VN request failed
        Discard $G_s'(V_s', E_s')$ and temporary node mapping array
    else
        Setup successful
        Reserve mapped node resources (σc^i) from node mapping array onto $G_s(V_s, E_s)$
        Reserve routed link resources (σb^i)onto $G_s(V_s, E_s)$
    end if
else
    Setup successful
    Reserve mapped node resources from node mapping array onto $G_s(V_s, E_s)$
    Reserve routed link resources onto $G_s(V_s, E_s)$
end if

V. Performance Analysis

The performance of VN scheduling algorithms is now tested. Namely, discrete event simulation models are developed using the *OPNET* $Modeler^{TM}$ toolkit to generate and process VN requests/demands. Tests are then done using a modified NSFNET physical network substrate with a moderate node degree of 3.12 (i.e., 16 nodes, 25 links), see Figure 1. All nodes are assumed to have 1,000 units of generic resource capacity, and all physical links have 10,000 units of bandwidth. Meanwhile, VN requests are generated by composing random graphs with 4-7 nodes each, with an average node degree of 2.6. The requested VN node capacities are uniformly-distributed between 10-30 units, and VN link capacities are distributed between 100-1,000 units. Accordingly, $\mathbb{B}$ is set to 1,000, $\mathbb{C}$ is set to 30, and both ρ and π are set to unity in Eqs. 1 and 2. Also, for the PBR scheme, up to three nodes/links are assigned higher priority levels, see Figure 2. Meanwhile, for the CBR scheme, a median value of $\sigma = 0.5$ is chosen for all tests.

Finally, all VN requests have exponentialy-distributed holding and inter-arrival times, with means μ and λ, respectively. For testing purposes, a scaled mean holding time of $\mu = 10$ seconds is used here, and the mean inter-arrival times are adjusted according to load. Note that these values are relative numbers and do not necessarily reflect real-world timings. Finally, a modified Erlang load metric is also defined as:

$$\text{Modified Erlang load} = \sum_{n=4}^{7} (n-1) \times \mu/\lambda \qquad (3)$$

i.e., where VN request sizes range from n=4-7 nodes, and $1/\lambda$ is the mean inter-arrival rate (requests/sec).

Blocking results are first presented in Figure 3. As expected, both the PCR and CBR schemes give lower failure rates as compared to the base Algorithm 1. Furthermore, the PBR scheme generally gives the lowest blocking out of all three schemes, i.e., about 40% lower

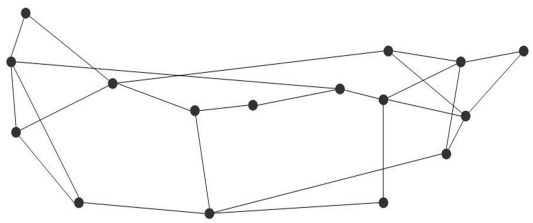

Fig. 1. Test network topology

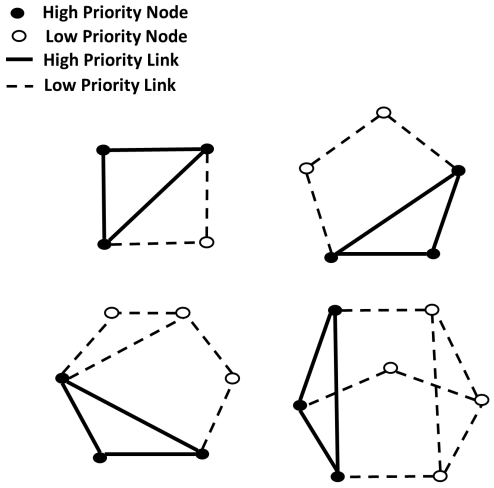

Fig. 2. Prioritized VN requests

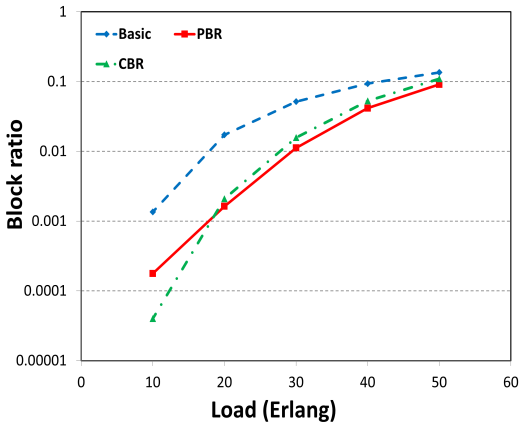

Fig. 3. VN request blocking rate

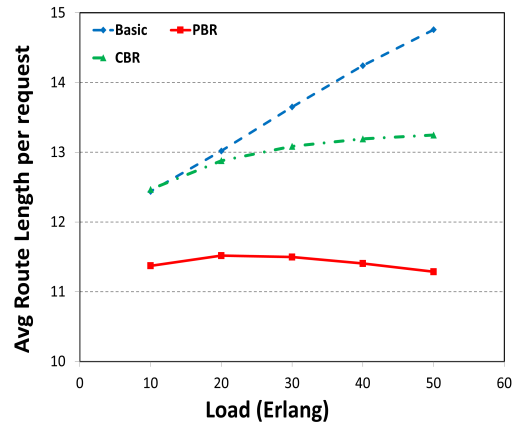

Fig. 4. Average route length per request

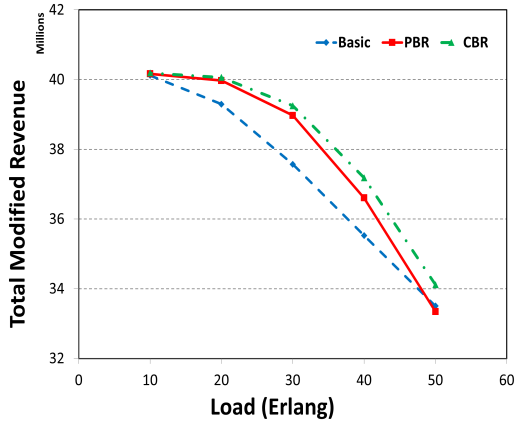

Fig. 5. Modified revenue

than the baseline scheme at higher loads. However, the CBR scheme is quite competitive here and even outperforms the PBR approach at lower loads. Next,

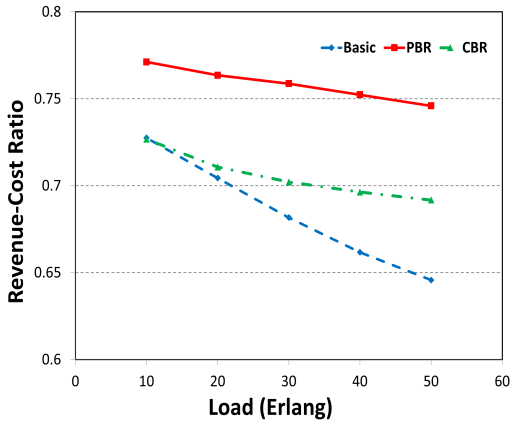

Fig. 6. Revenue-cost ratio

the average path lengths for the mapped VN links are also shown in Figure 4. Since the PBR scheme may only map a fraction of the VN links here, this approach gives the shortest path lengths. Conversely, both the baseline Algorithm 1 and the CBR scheme tend to give longer path lengths, with the latter giving lower usage at higher loads. Overall the ability to compromise between requested resources and durations gives the CBR approach more freedom to allocate VN nodes and route VN links, especially at higher loads.

Finally, the total revenue and revenue-cost ratios are plotted in Figures 4 and 5, respectively. Here, both partial provisioning strategies give better performance versus the baseline Algorithm 1. Moreover, the PBR solution yields slightly higher revenue-cost ratios than the CBR scheme. However, more tests are needed to gauge the relative performance of these schemes for different numbers of high/low priority nodes (PBR scheme) and varying σ values (CBR scheme).

VI. Conclusions and Future Work

Cloud-based service demands are driving the need for virtual network reservation. However, existing network reservation studies have not addressed this problem area. Hence this paper presents a novel baseline scheme for scheduling virtual network requests and also develops two further variations to provision partial demands. Overall simulation results show good improvements in terms of blocking ratios and revenues. Future efforts will study more advanced service configurations with both advance reservation and immediate reservation demands.

Acknowledgment

This work was made possible by the NPRP 5-137-2-045 grant from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

REFERENCES

[1] J. Zheng, *et al*, "Routing and Wavelength Assignment for Advance Reservation in Wavelength-Routed WDM Optical Networks", *IEEE International Conference on Communications*, New York City, NY, April 2002.

[2] N. Charbonneum, V. Vokkarane, "A Survey of Advance Reservation routing and Wavelength Assignment in Wavelength-Routed WDM Networks", *IEEE Communications Surveys and Tutorials*, Vol. 14, No. 4, 2012, pp. 1037-1064.

[3] F. Gu, *et al*, "Virtual Overlay Network Scheduling", *IEEE Communication Letters*, No. 8, 2011, pp. 893-895.

[4] R. Guerin, *et al*, " Networks with Advance Reservations: The Routing Perspective", *IEEE INFOCOM*, Tel Aviv, Israel, March 2000.

[5] J. Zheng, *et al*, "Towards Automated Provisioning of Advanced Reservation Service in Next-Generation Optical Internet", *IEEE Communications Magazine*, Vol. 44, No. 12, 2006, pp. 68-74.

[6] D. Andrei, *et al*, "Integrated Provisioning of Sliding Scheduled Services Over WDM Optical Networks", *IEEE/OSA Journal of Optical Comm. Networks*, Vol. 1, No. 2, 2009, pp. A94-A105.

[7] T. Wallace, A. Shami, "Connection Management Algorithm for Advance Lightpath Reservation in WDM Networks", *IEEE Broadnets 2007*, Raleigh, NC, September 2007.

[8] L. Shen, *et al*, "A Two-Phase Approach for Dynamic Lightpath Scheduling in WDM Optical Networks", *IEEE ICC 2007*, Glasgow, Scotland, June 2007.

[9] C. Xie, *et al*, "Rerouting in Advance Reservation Networks", *Computer Communications*, Vol. 35, 2012, pp. 1411-1421.

[10] M. Chowdhury, *et al*, "ViNEYard: Virtual Network Embedding Algorithms With Coordinated Node and Link Mapping", *IEEE/ACM Transactions on Networking*, Vol. 20, No. 1, 2011, pp. 206-219.