Poster: Constraint-driven Virtual Network Design on a Shared Substrate

Jing Lu (student) and Jonathan Turner (faculty)
Department of Computer Science and Engineering
Washington University in St. Louis
St. Louis, MO 63130
Email: {jl1, jst}@arl.wustl.edu

In recent years there has been a growing recognition that the protocols at the heart of the Internet have become so resistant to change that practical progress in networking has stalled. Virtualization has been proposed as a vehicle for overcoming the growing problem of internet ossification [1]. In a virtualized network infrastructure, diverse virtual networks share a common physical $substrate$ consisting of both links and flexible network platforms capable of hosting multiple virtual routers [2]. Geographically distributed end users reachable through the substrate network have the freedom to choose a virtual network which offers services that best meet the users' requirements. Such a diversified Internet not only makes it easy to deploy new network architectures and technologies, but also provides a competitive environment for stimulating innovation and higher value services.

A substrate network can be represented by an undirected graph, in which each edge has a bandwidth capacity and an associated length. Each node can be viewed as a traffic aggregation point for the users to access a particular virtual network. Given a large-scale substrate network that reaches millions of potential users, it is important to develop efficient methods that enable virtual network designers to design virtual networks on top of the substrate. As a result, the virtual network should be able to support any traffic pattern allowed by a general set of traffic constraints with minimal network cost. In this paper, we present an efficient algorithm that has been implemented as an automated design tool to assist virtual network designers in finding the least-cost virtual network configuration on any given substrate network. Through experiments, we show some fundamental results relating the least-cost virtual network topologies and the traffic constraints. The algorithm is shown to be efficient in producing high quality solutions to practical-sized problems.

In this paper, we focus on the system contexts in which substrate resources can be reserved for use by different virtual networks, and where resource requirements are defined in terms of a set of general traffic constraints. An assumption we have made throughout this study is that the substrate links have sufficient capacities to handle the traffic for the virtual network being mapped. This is typically true when no single virtual network uses a large fraction of the substrate capacity and no substrate is operating at high usage loads.

Our research is built upon prior work on the constraint-based network design [3]–[5]. In general, a constraint can be stated informally as "the traffic from node set A to node set B is at most C." Specifically, we consider three classes of traffic constraints that we believe to be adequate in describing traffic requirements: *ingress/egress constraints*, often called the host model [6], that specify the total traffic entering and leaving each node; *pair-wise constraints* that define the traffic flows between each pair of nodes as a matrix; and *distance-based constraints* that bound the amount of traffic between each node and its more distant peers.

Given a set of traffic constraints, the objective of constraint-based virtual network design is to find a network configuration that can handle any traffic pattern allowed by the constraints. This involves selecting a network topology comprising virtual links and virtual routers. Finding the best topology is a hard problem by itself. Therefore, we explore a restricted form of virtual network topology that we call *backbone-star* topology, in which some of the nodes are designated as backbone nodes, while the remainder are referred to as access nodes. Each access node has a single edge connecting it to a backbone node, meaning that each backbone node is at the center of a "star" formed by its neighboring access nodes. The subset consisting of the backbone nodes are connected together to form the backbone topology. In a selected virtual network topology, virtual links are typically provisioned under the assumption that traffic is routed along the least-cost path, although other routing policies are also used. (It should be noted that the routes used for network dimensioning are best viewed as the "default paths" rather than the only paths that may be used in the operational network.)

Dimensioning virtual links to have sufficient capacity starts with computing the shortest path on a selected virtual network topology. However, shortest paths can not be computed without knowing the lengths of the virtual links, which requires knowledge of the mappings of virtual nodes onto the substrate. This inter-dependency of mapping virtual nodes and link dimension has led us to adopt an iterative method. Its goal is to find a configuration that minimizes the overall link costs where the cost of a virtual link is proportional to the product of its capacity and its physical length. Our algorithm is outlined below.

1) *Choose a candidate substrate network.* A virtual network designer picks a substrate network from many

candidates based on its requirements and other factors such as leasing price and substrate services, etc.

2) *Select access nodes on the substrate.* Depending on what regions the virtual network is to cover with the service and with consideration of management overhead and costs, the virtual network designer may select all substrate nodes or just a subset of them to be included in the virtual network as access nodes to the users.
3) *Decide a backbone topology and an initial mapping of backbone nodes onto the substrate*. In this step, the virtual network designer decides the number of backbone nodes and the way they are interconnected. The initial mapping of the backbone nodes can be arbitrary and simply provides a starting point for the iterative refinement procedure.
4) *Connect access nodes to the closest backbone nodes and map virtual links to substrate paths*. To provide flexibility in the virtual network topology, we do not make a rigid connection between access nodes and backbone nodes. Instead, during each iteration, we connect each access node to the backbone node that is closest to it in the substrate. Then each virtual link is mapped to the shortest path between the two substrate nodes, where the two ends of the virtual link are mapped.
5) *Determine link capacities*. Given a complete virtual network topology with known link lengths, we can compute the shortest paths in the virtual network. Along with the traffic constraints defined, link capacity can then be dimensioned using linear programming as in [4]. For the three classes of traffic constraints considered here, we formulate the dimension of each virtual link to an equivalent maximum flow problem to enable a much faster solution [7].
6) *Find best backbone node mapping*. The previous steps result in a complete virtual network topology with defined link capacities. We now explore alternative mappings of backbone nodes onto the substrate, while maintaining the same virtual backbone topology and link capacities. We formulate the backbone mapping problem as a Mixed Integer Quadratic Program (MIQP) and solve it efficiently using [8]. The best mapping found in this step is then used in the next iteration of the algorithm, which continues from step 4.

In each iteration, the virtual network configuration is evaluated at step 5 in terms of the total link costs. The iterative procedure terminates either when there is no further improvement in the quality of the solution obtained, or a pre-specified upper bound on the number of iterations is reached. We have developed a tool that automates the iterative design process.

We investigate, under a wide range of traffic conditions, the relative cost-effectiveness of different backbone topologies on three substrate networks consisting of 20 to 50 of the largest metropolitan cities in the US and western Europe [5]. We consider five different backbone topologies: complete graph, ring, star, star ring and minimum spanning tree, and study how the most cost-effective topology changes as the tightness of pairwise traffic constraints and the constraints on traffic locality are varied. Our experiments show that the system of traffic constraints has a profound influence on the least-cost network structure. In particular, tight pairwise constraints favor network topologies in which all pairs of nodes are directly connected by links with just the right capacity. As constraints get looser, "tree-like" topologies are advantageous in reducing the network costs. With constraints provided by the pure hose model, the most cost-effective network topologies turn out to have all nodes connected through a single, centrally located backbone node. In addition, we find that the least-cost network structure is also affected by the underlying substrate network topology. To get a sense of how close the least-cost configuration found by our tool is to the optimal design, we compute a lower bound on the network cost for each substrate with varying traffic constraint parameters. Overall, the cost of the constructed virtual networks using the iterative algorithm is usually no more than 1.5 times the lower bound and the quality of solutions improves to within 10% of the lower bound as the traffic locality gets weaker.

The running time of an iteration is dominated by the time to solve an MIQP. Mapping time is a function of the substrate network topology, the virtual backbone topology and the number of backbone routers in the virtual networks. In our experiments, mapping a virtual network with 16 backbone routers and 50 access nodes on a substrate consisting of 50 US cities takes 1.7 seconds for the ring topology and less than 0.4 second for the star and complete topologies. The iterative algorithm also converges fairly quickly. Even though all initial backbone placements are randomly selected, nearly all experiments end within 6 iterations on substrates with 20 cities, and more than 96% of the experiments finish within 9 iterations on the substrates with 50 cities. Overall, the iterative algorithm is capable of designing a high quality virtual network on a given substrate in a reasonably short time, which is critical for virtual network designers to deploy their services cost-effectively, and for substrate providers to manage and allocate available resources efficiently.

REFERENCES

[1] L. Peterson, S. Shenker, and J. Turner, "Overcoming the Internet impasse through virtualization," in *ACM Workshop on Hot Topics in Networks (HotNets)*, 2004.

[2] J. Turner and D. Taylor, "Diversifying the Internet," in *IEEE GLOBECOM*, 2005.

[3] A. Fingerhut, S. Suri, and J. Turner, "Designing Least-Cost Nonblocking Broadband Networks," *Journal of Algorithms*, pp. 287–309, 1997.

[4] A. Fingerhut, "Approximation Algorithms for Configuring Nonblocking Communication Networks," *Doctoral Dissertation, Washington University in St. Louis*, May 1994.

[5] S. Y. Choi, "Resource Configuration and Network Design in Extensible Networks," *Doctorial Dessertation, Washington University in St. Louis*, Dec. 2003.

[6] N. G. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramakrishnan, and J. E. van der Merive, "A flexible model for resource management in virtual private networks," in *ACM SIGCOMM*, 1998, pp. 95–108.

[7] J. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *J. Assoc. Comput. Mach.*, vol. 19, pp. 248–264, 1972.

[8] R. Fletcher and S. Leyffer, "A Mixed Integer Quadratic Programming Package," *University of Dundee*, 1998.