Attachment A: A Decomposition Approach to the Virtual Network Embedding (VNE) Problem

I. Virtual Network Embedding (VNE) in Network Virtualization

From Virtualization to Network Virtualization: The essence of virtualization is the abstraction and pooling of physical resources to enable resource sharing among multiple users. For instance, Amazon EC2 service [1] deploys thousands of powerful servers (of abundant resources such as CPU, memory, and hard disks). Those server resources are sliced through a virtualization software (namely hypervisor), and each slice forms an independent virtual machine (VM) that can be leased to a tenant (e.g., Netflix, and Toyota) [1]. The benefits of virtualization are multi-fold. First, the tenant (e.g., Toyota) can obtain a running VM to host a website with just a few mouse-clicks (i.e., agile deployment). The VM functions no different from a physical server, and is under the tenant's sole ownership. Second, tenant can add/remove/upgrade VMs based on run-time needs (e.g., during the Super Bowl week, Netflix may upgrade the rented VM to support increased website visits.). Third, the maintenance, security, deployment, upgrading of physical servers are oblivious to tenants (e.g., a tremendous cost reduction), and can be centrally provided by the domain experts of Amazon (i.e., a technical vantage). Furthermore, in virtualization, a tenant may rent multiple VMs, and expect to *connect the VMs as a network of their own*. This results in the idea of *network virtualization*, where not only servers are virtualized as VMs, but also bandwidth resources on links are virtualized to connect VMs. In other words, each user can obtain a sliced and isolated portion of the bandwidth resources to form a path that connects the VMs of his or her own.

What is Virtual Network Embedding? In network virtualization, a user typically expresses the demand as a virtual network request (VNR) that consists of virtual nodes (that represent the needed VMs) and virtual links (that represent the required connections between VMs). For instance, a global company needs to deploy three computing servers located in Asia, Europe, and North America to serve customers in respective continents. In Computer Networking, this service feature is modeled as a virtual network as shown in Fig. 1, which contains three virtual nodes (representing the need for three (virtual) computing servers), and three virtual links (indicating that servers should be logically connected to each other). After receiving this request, Amazon can explore the physical network in Fig. 2 consisting of nine computing servers that are connected via eleven physical links. One possible offer is to map the three virtual nodes to (circled) physical servers located in China, Sweden and the USA, respectively, and connect each pair of servers with a mapped path. Although this decision appears to be simple, it can be very hard to reach in practice since Amazon generally has a large server pool (e.g., only nine above), hence leading to numerous combinations when picking up hosting servers (i.e., only three for above example) as well as numerous choices of mapped paths. Among all of the combinations, the best or optimal offer should take the resource availability, location, and cost into consideration since all the resources are priced. In fact, above general process that allocates physical resources to a given service request is called Virtual Network Embedding (VNE) problem [2], which contains two decisions: (i) Decide the hosting physical server to create a VM for each virtual node; (ii) Decide the physical (mapped) path for connecting VMs according to each virtual link. Unfortunately, the VNE problem is shown to be NP-Hard, which is a type of problem that is extremely hard to solve efficiently for large instances [3].

.

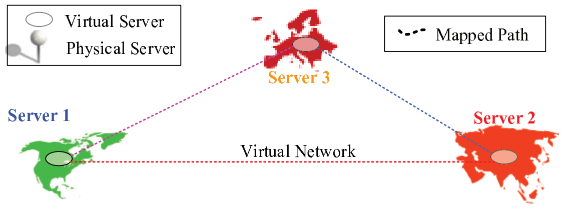


Fig. 1. VNR

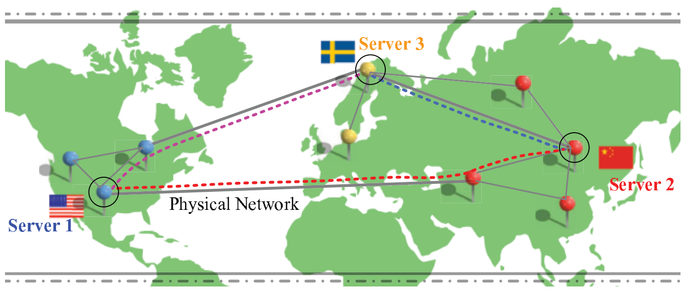


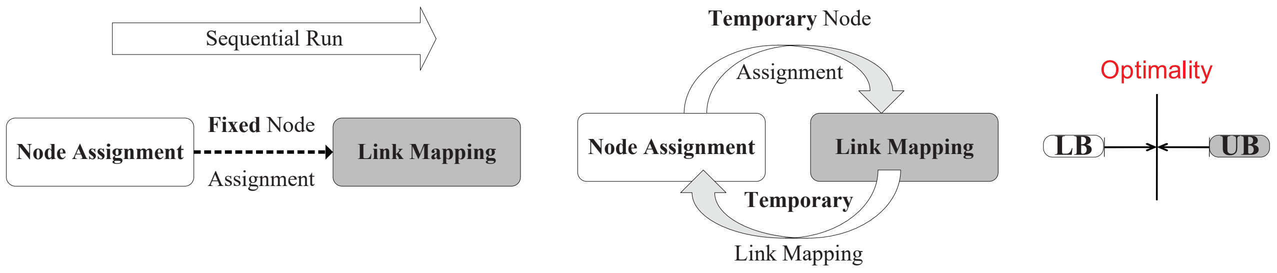
Fig. 2. VNE

II. Plan of Study in Our Project

In Computer Science, when addressing a challenging problem, one common strategy is *divide-and-conquer* where the complex structure of the original problem is decomposed into manageable pieces that can be easily addressed. This idea has been applied to the VNE problem, resulting in two sub-problems: Node Assignment (NA) that places each virtual node as a VM on a physical machine; and Link Mapping (LM) where paths are created between VMs for connections. This idea, endorsed by most existing literature, is abstracted in Fig. 3(a) below where the VNE is addressed in two sequential phases. In the first phase, the node assignment is determined (e.g., VMs are placed for the three computing servers in respective regions of Fig. 2) *without the consideration of link connections*; in the second phase, as all the VMs are already fixed at a given location, one can easily find a path to connect the respective VMs. The simplicity of above strategy, however, trade-offs the optimality of the VNE solution (due to the *nearsightedness* of the first phase which is oblivious to the link mapping in the second phase). For instance, in the first phase, a computing server is possibly be placed in a distant (cheap-priced) location which turns out to be very costly to connect to in the second phase.

In this project, we strike to *maintain the simplicity of this* divide-and-conquer *strategy without sacrificing the optimality of the VNE solution*. We targets at a idea-wise-simple but mathematically-sound solution, as abstracted in Fig. 3(b). Being aware of the sub-optimality of sequential divide-and-conquer solutions (Fig.3 (a)), we introduce a feedback process that allows the temporary NA and LM decisions to evolve based on the feedback from each other. In other words, node assignment that is not wisely done at first can be tuned in later iterations with the feedback from the link mapping, and vice versa. In fact, we can prove that the temporary solutions to the NA, LM sub-problems lead to lower bound (LB) and upper bound (UB) of the original problem (Fig. 3(c)), this process can terminate when the two bounds meet or are close enough, which indicates that optimal or near-optimal solution is found.

Note that above examples and associated discussion are intended to keep simple for the ease of reading. In reality, the VNE problem can be very challenging: the size/topology/resource availability of the virtual network and physical network can both vary, and the decision may need to be done in a short window (i.e., on-line case) for large network instances. Our first step in this project is to develop a basic sequential divide-and-conquer model for the VNE problem, which serves as a benchmark. In the second step, based on the basic model, we plan to apply decomposition approach to create path-based models for the two sub-problems (i.e., NA, and LM), and establish the feedback in-between. In the final step, we plan to resolve above models with a state-of-the-art ILP solver software - *IBM CPLEX* to find optimal/near-optimal solutions. Overall, we expect the findings in this project fill up one important gap in the literature (i.e., the lack of performance guarantee in VNE solutions based on *divide-and-conquer*), and shed lights on researches of a family of variations of VNE problems in network virtualization.



(a) Sequential Divide-and-Conquer (b) Divide-and-Conquer with Feedback (c) Bounds

Fig. 3. Divide-and-Conquer Strategies

References

[1] EC2, "https://aws.amazon.com/ec2/customers/."
[2] Y. Wang, Q. Hu, and X. Cao, "A branch-and-price framework for optimal virtual network embedding," *Comput. Netw.*, vol. 94, no. C, pp. 318–326, Jan. 2016.
[3] Y. Wang, Q. Hu, C. Li, J. Flor, and M. Jalalitabar, "Node re-visitation in virtual service mapping: An integrated approach," in *2021 IEEE/CIC ICCC*, 2021, pp. 887–892.