Virtual Network Embedding with Substrate Support for Parallelization

Sheng Zhang, Nanjing University, P.R. China
Jie Wu, Temple University, USA
Sanglu Lu, Nanjing University, P.R. China

Presenter: Pouya Ostovari, Temple University, USA

IEEE Globecom 2012
Anaheim, CA,
3-7 Dec, 2012

Agenda

- Introduction
- Motivation for parallelization
- Virtual network embedding – parallelization version
- Two Algorithms
- Simulations
- Conclusions

The Internet has proven its worth by

- Improving the way we access and exchange information in the modern world
- Supporting multitude of distributed applications and a wide variety of network technologies

However, like many successful technologies
the Internet is suffering its adverse effects

Internet Ossification

Alternations to the Internet architecture are restricted to simple incremental updates,

e.g., the deployment of IPv6

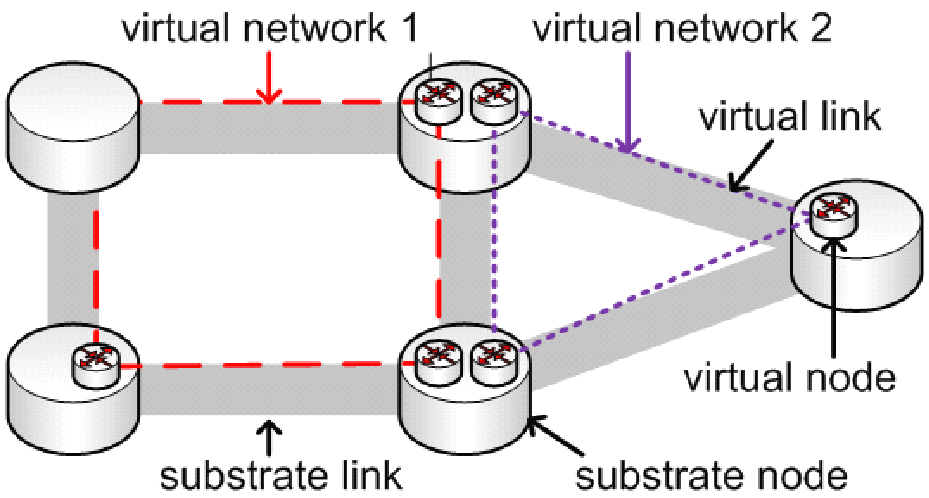
- Multiple network domains with conflicting interests
  - multilateral relationship? Difficult!
  - Deploy changes/updates? Global agreement!
- The

  Intern
  - security, routing st

Flexibility + Diversity

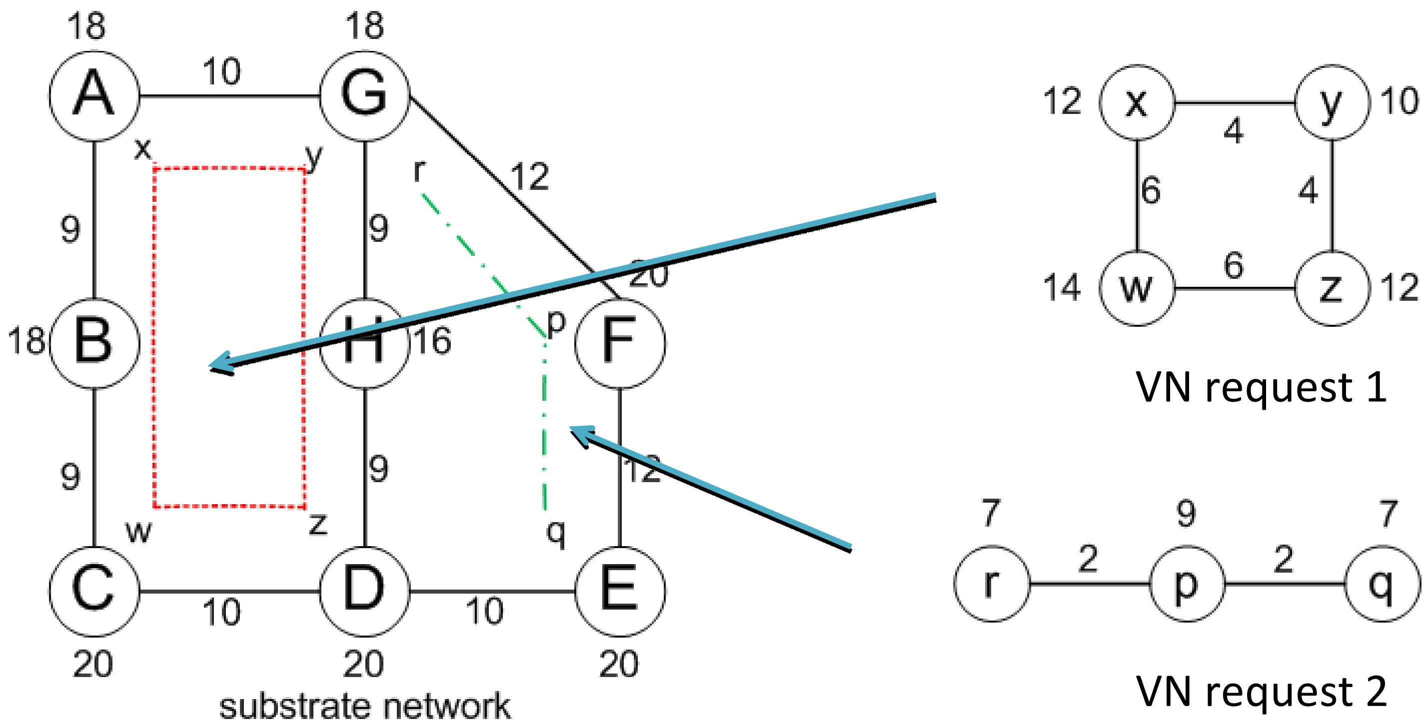
Network Virtualization

- Infrastructure provider (InP)
  - Maintains physical/substrate network (SN)
- Service provider (SP)
  - purchases slices of resource (e.g., CPU, bandwidth, memory) from the InP
  - then creates a customized virtual network (VN) to offer value-added service (e.g., content distribution, VoIP) to end users
- End users



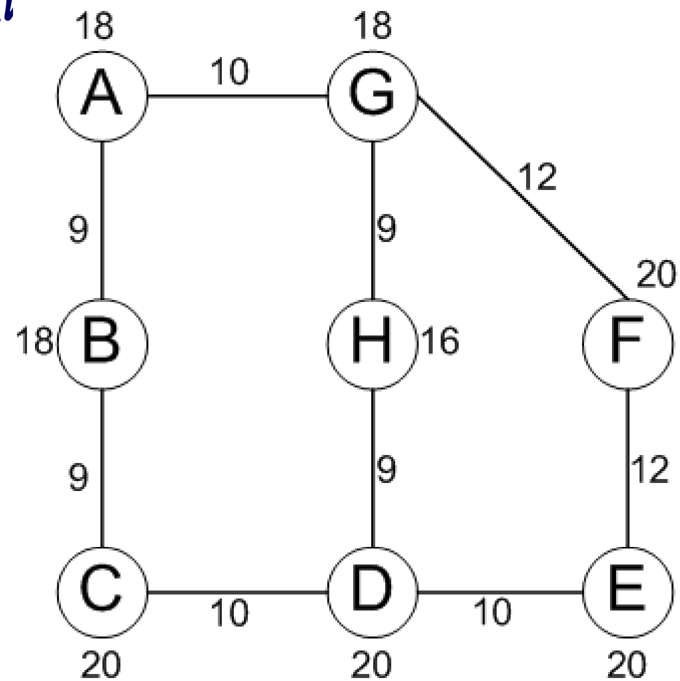
Virtual Network Mapping

- VNM is to embed multiple VN requests with resource constraints into a substrate network
- The objective is usually to maximize the utilization ratio of physical resources

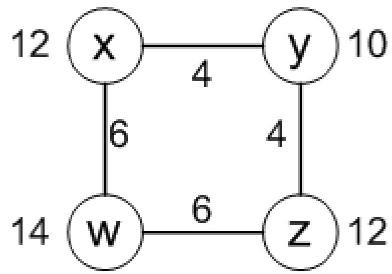


Virtual Network Mapping

Given a VN request and a substrate nerwork, the problem of determining whether the request can be embeded without any resource violation is NP-hard
[Andersen 2002]



substrate network



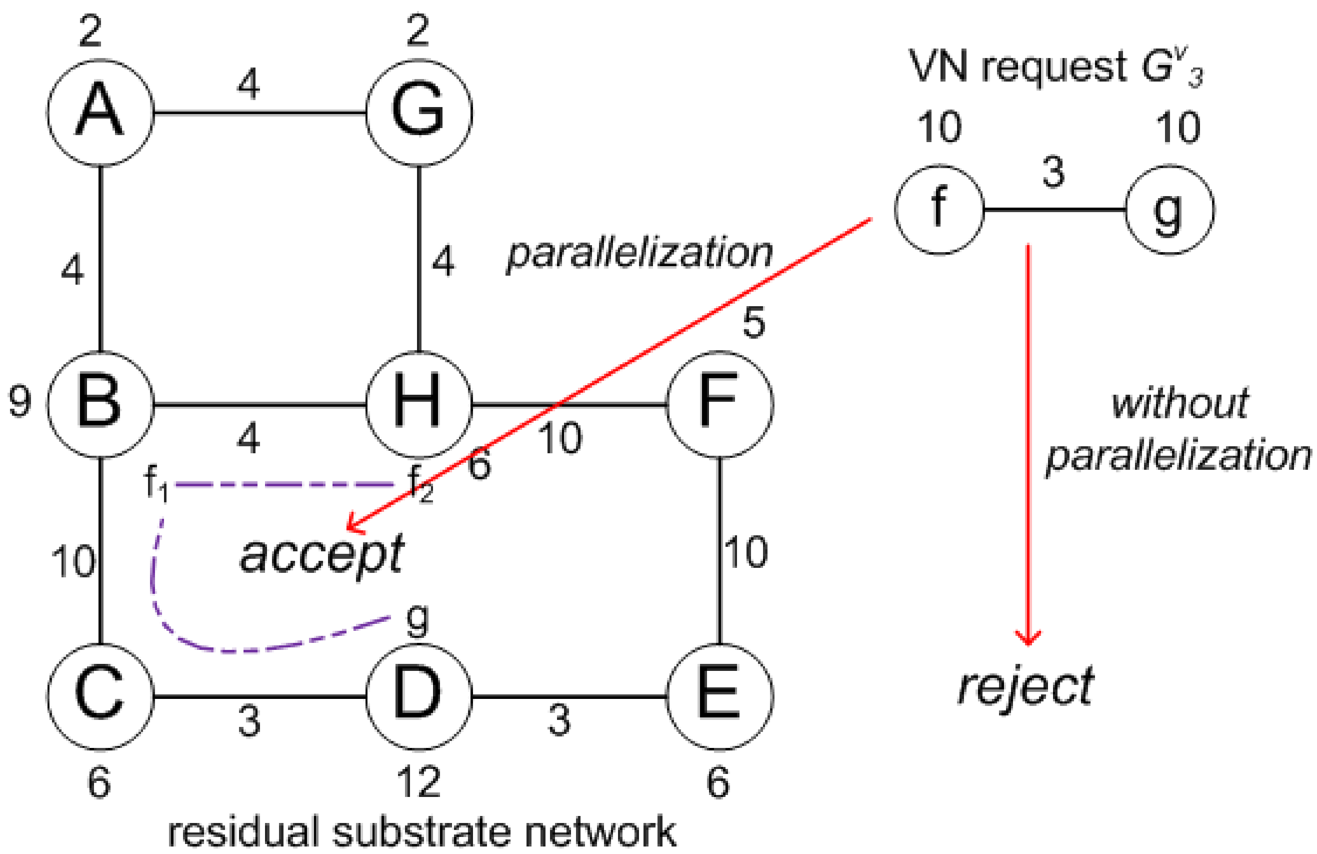
Related Work

- Simulated annealing: [Ricci et al. 2003]
- Load balancing: [Zhu & Ammar 2006]
  - Unlimited resources
- Path splitting: [Yu et al. 2008]
  - Multi-commodity flow problem
- Opportunistic resource sharing: [Zhang et al. 2011&2012]
- Topology-aware: [Cheng et al. 2011][Zhang et al. 2012]

However

- There has been little research done on virtual network embedding with substrate support for parallelization
  - SN allows a virtual node to be mapped to multiple substrate nodes
- Advantages
  - Makes substrate resource utilization more efficient
  - Makes virtual networks more reliable, as computation can be quickly migrated to other substrate nodes in case a substrate node crashes.

A Motivational Example



How to capture the parallelization?

- The multiple substrate nodes, which one virtual node is mapped onto, should be close to each other so as to mitigate the effect of network latency.
- This paper represents this ``closeness'' by forcing these multiple substrate nodes to form a star topology, i.e., all *slave* nodes are 1-hop away from the *master* node.

Virtual network embedding - the parallelization version

- Virtual network embedding from a virtual network to a subset of a substrate network is composed of three components:
  - *Master mapping*
    - A virtual node to a substrate node
  - *Slave mapping*
    - A virtual node to a group of substrate nodes
  - *Link mapping*
    - A virtual link to a substrate path
- The objective is to maximize the utilization ratio of substrate resources

A Concrete Example

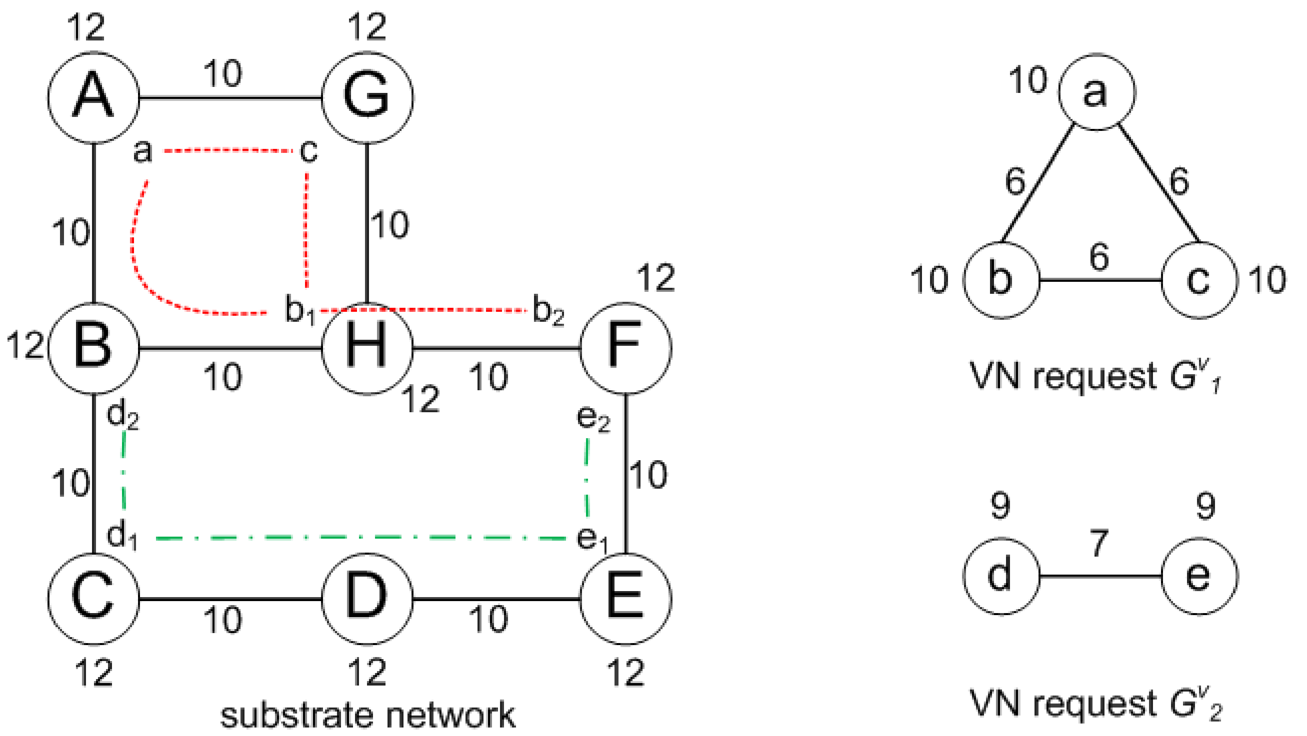


Fig. 1. An illustration of virtual network embedding with substrate support for parallelization. For virtual network G_1^v, the master mapping is $\{a \to A, c \to G, b \to H\}$, the slave mapping is $\{a \to \emptyset, c \to \emptyset, b \to \{F\}\}$, and the link mapping is $\{(ac) \to \{AG\}, (cb) \to \{GH\}, (ba) \to \{HB, BA\}\}$. $Ratio(b) = \{0.6, 0.4\}$. For virtual network G_2^v, the master mapping is $\{d \to C, e \to E\}$, the slave mapping is $\{d \to \{B\}, e \to \{F\}\}$, and the link mapping is $\{(de) \to \{CD, DE\}\}$. $Ratio(d) = \{2/3, 1/3\}$, and $Ratio(e) = \{2/3, 1/3\}$.

The big picture of ProactiveP

- ProactiveP employs a greedy approach to deal with master mapping.
- The slave nodes are chosen from the neighbors of each master node.
- The link mapping utilizes Dijkstra to find the shortest path that meets the bandwidth requirements.

ProactiveP

- *Initialization: every substrate node is set to be unused, and every node updates $RC^{s}_{nei}(n^{s})$, denoting the summation of the residual resources of the neighbors (including n^{s} itself) of n^{s}*
- *Master Mapping: all virtual nodes are sorted in the decreasing order of $C^{v}(n^{v})$. We then map each virtual node to the unused substrate node with the largest $RC^{s}_{nei}(n^{s})$*

ProactiveP

- Slave Mapping: all the neighbors of a master node are chosen as the slave nodes. Then the CPU requirement is divided into pieces that are proportional to the residual units of CPU in neighbors of the master node.
- Link Mapping: each virtual link is mapped to the shortest substrate path that satisfies the bandwidth requirement between the corresponding endpoints

LazyP

- LazyP shares most parts with ProactiveP, except the slave mapping phase,
- LazyP applies parallelization only when the residual units of CPU in the master node for a virtual node is not sufficient.
  - when there is a need for parallelization LazyP iteratively chooses the unused node that has the most residual units of CPU among the neighbors of the master node for a virtual node, and tries to satisfy the CPU demand until the virtual node is successfully embedded.

Simulation Setup

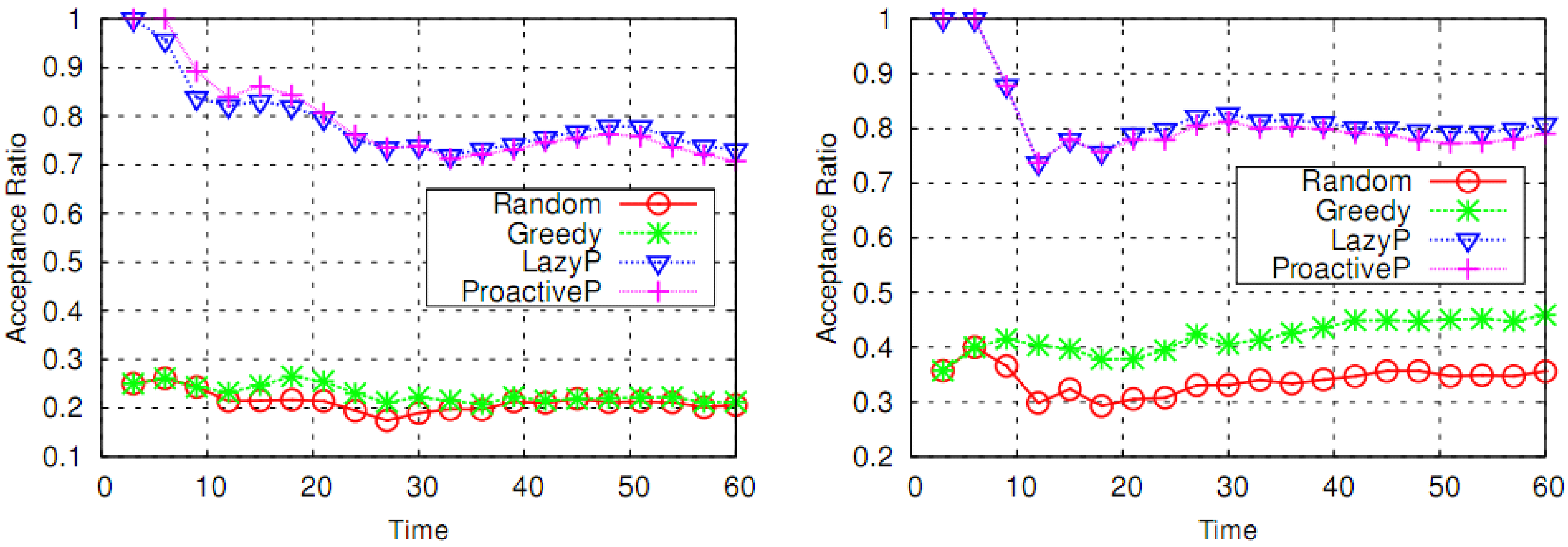
Similar settings to several existing studies

- Substrate network
  - Topology: Arpanet & Erdos-Renyi Graph (20,0.4)
  - CPU & Bandwidth: [50,100], uniform
- Virtual network
  - # of nodes: [2,10], uniform
  - Each pair of nodes connects with probability 0.5
  - Lifetime: 10 minutes, exponential
  - Arrivals: Possion process (0.2 minutes)

Simulation Setup

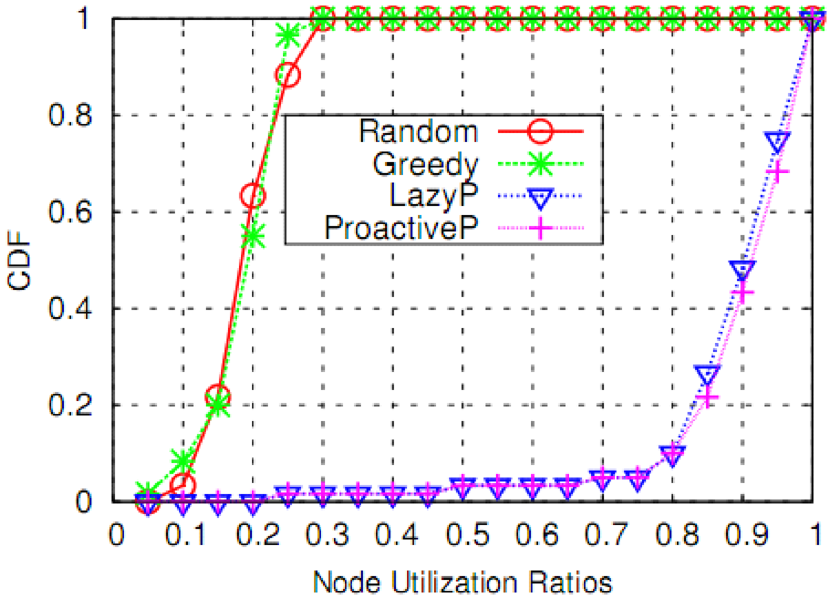
- Performance metrics
  - Acceptance ratio: the higher, the better
  - Node/link utilization: the higher, the better
- Algorithms in comparison
  - *ProactiveP*
  - *LazyP*
  - Random: node mapping is randomly generated
  - Greedy: node mapping is greedily generated

Simulation Results – Acceptance Ratio

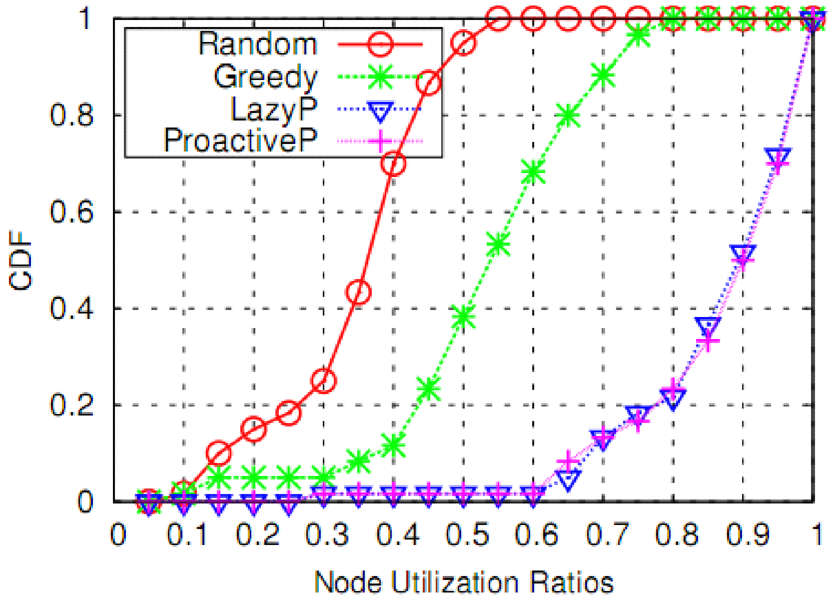


(a) On *ArpaNet* graph (b) On *Erdős-Rényi model* $G(20, 0.4)$

Simulation Results – Node Utilization

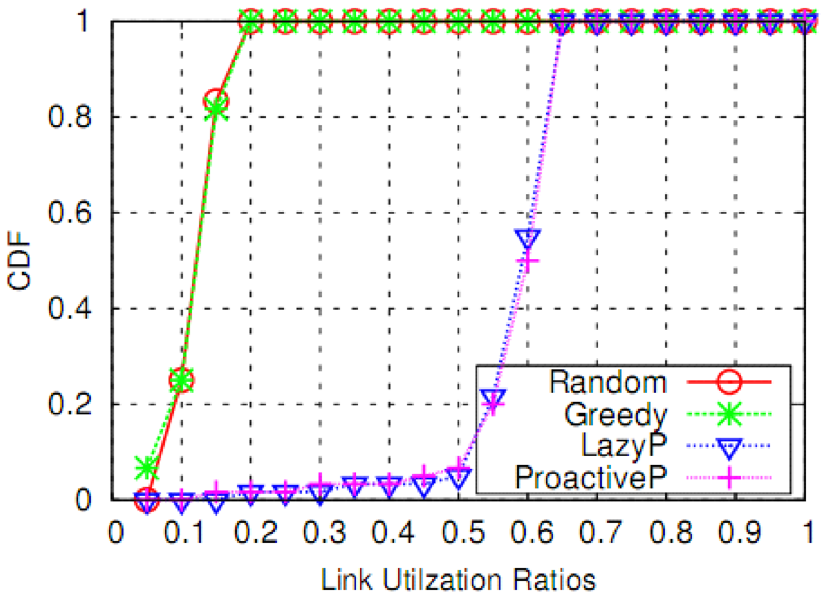


(a) On *ArpaNet* graph

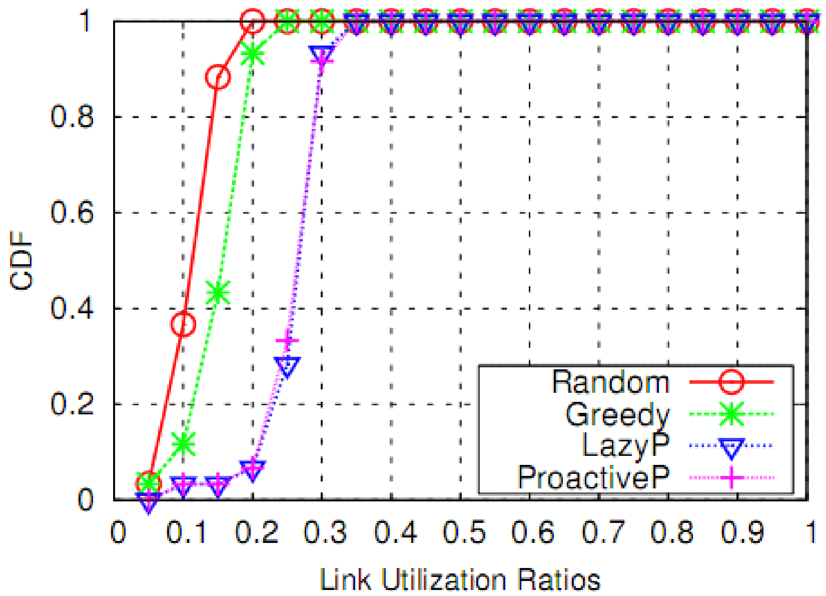


(b) On *Erdős-Rényi model* $G(20, 0.4)$

Simulation Results – Link Utilization



(a) On *ArpaNet* graph



(b) On *Erdős-Rényi model* $G(20, 0.4)$

Conclusions

- To explore how we can design the substrate network to best serve the goals of network virtualization, we envisions that the substrate network supports parallelization.
- We formulate the parallelization version of the virtual network embedding problem, for which we develop two algorithms and three extensions.
- In future work, we intend to look in detail into this problem and combine path splitting with parallelization.

Thanks for your attention!
Q&A

http://cs.nju.edu.cn/disla
b/