On the Design of Reliable Virtual Networks

Raouf Boutaba

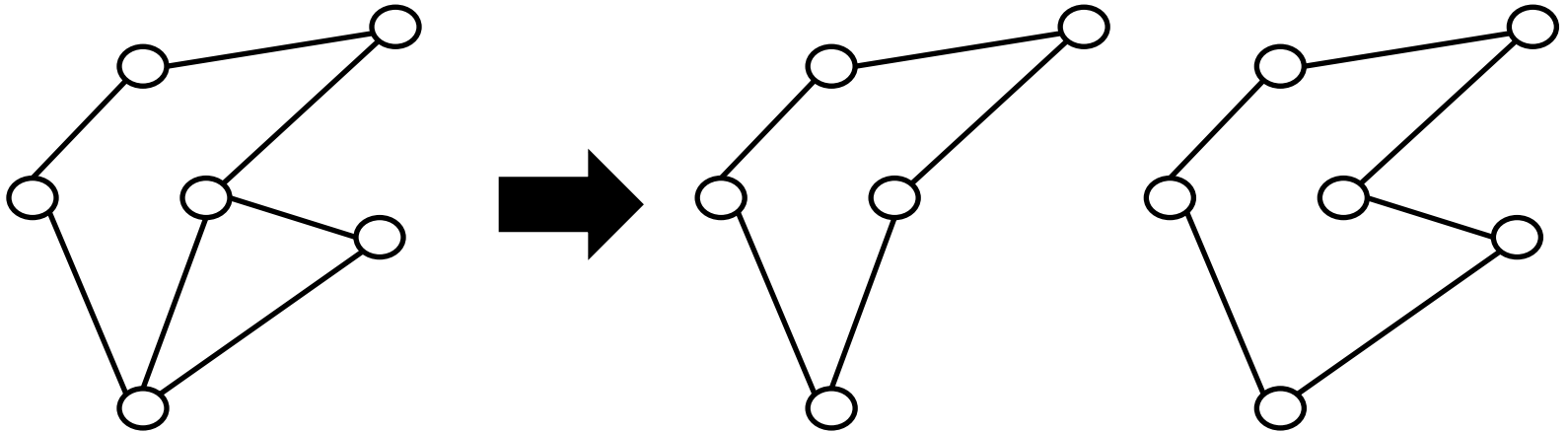
David R. Cheriton School of Computer Science
University of Waterloo

Outline

- Network Virtualization
  - Virtual Networks
  - Virtual Network Embedding
- Reliability in Network Virtualization
  - Survivable Virtual Network Embedding (SVNE)
- Recent Contributions
  - SiMPLE
  - CoVine
  - DRONE
  - VENICE
- Research Challenges/Directions

What is Network Virtualization ?

- Making a physical network appear as multiple logical ones



Physical Network

Virtualized Network - 1

Virtualized Network - 2

What is Network Virtualization?

- Transparent abstraction of networking platform and resources
  - Multiple logical interpretations of the physical characteristics
    - Multiple virtual networks (VNs)
- Additional level of indirection
  - Indirect access to network resources
- Resource partitioning and isolation
  - Physical and logical
  - Dynamic provisioning and configuration

Definition (Sort of)

Network virtualization is a networking environment that allows multiple service providers to dynamically compose multiple heterogeneous virtual networks that co-exist together in isolation from each other, and to deploy customized end-to-end services on-the-fly as well as manage them on those virtual networks for the end-users by effectively sharing and utilizing underlying network resources leased from multiple infrastructure providers.

M. Chowdhury and R. Boutaba. A Survey of Network Virtualization. Computer Networks (COMNET). Elsevier, Vol. 54(5), pp. 862-876, April 2010.

Why Network Virtualization?

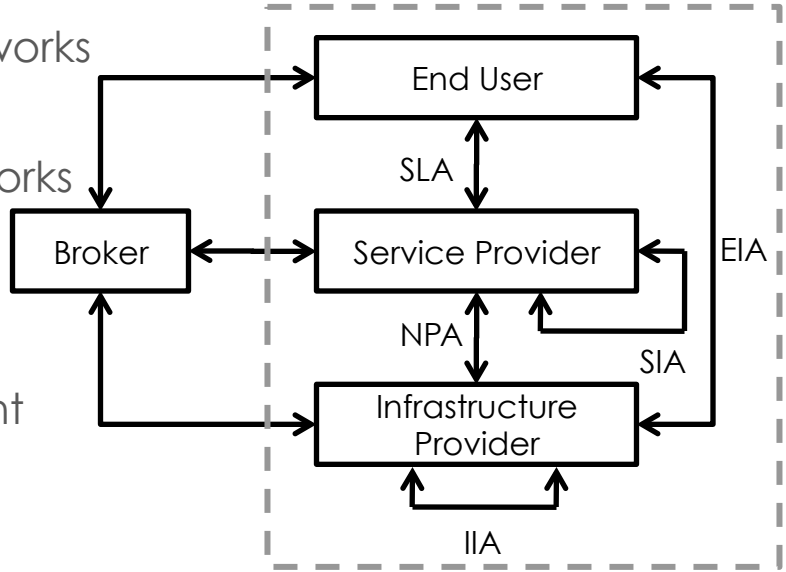
- Internet is *almost* ossified
  - Lots of band-aids and makeshift solutions (e.g., overlays)
  - A new architecture (aka clean-slate) is needed
- Hard to come up with a one-size-fits-all architecture
  - Almost impossible to predict what future might unleash
- Why not create an all-sizes-fit-into-one architecture instead!
  - Open and expandable
  - Coexistence of heterogeneous architectures
- Testbed for future networking architectures and protocols

Business Model

Players *

- Infrastructure Providers (*InP*)
  - Manage underlying physical networks
- Service Providers (*SP*)
  - Create and manage virtual networks
  - Deploy customized end-to-end services
- End Users
  - Buy and use services from different service providers
- Brokers
  - Mediators/Arbiters

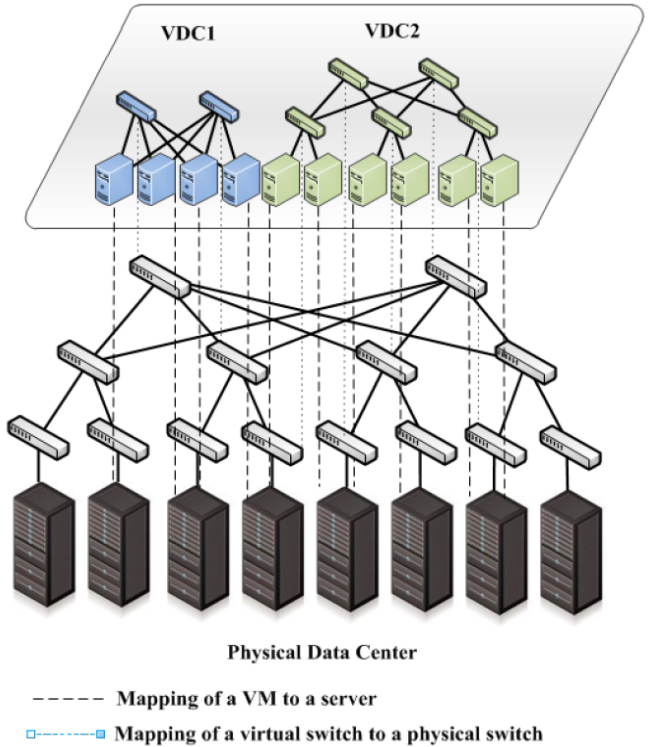
Relationships *



* M. Chowdhury and R. Boutaba. Network Virtualization: State of the Art and Research Challenges, IEEE Communications Magazine, 47(7): 20 – 26, July 2009

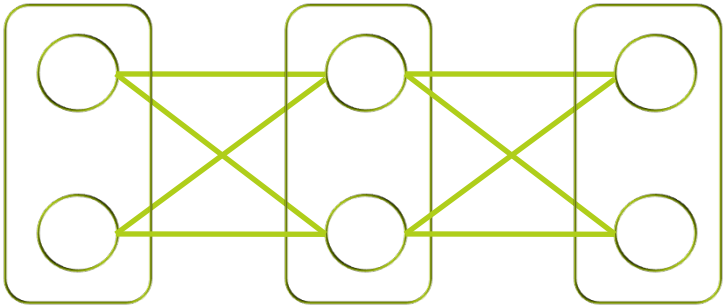
Recent Trend : Virtual Data Centers

- Cloud provides computing resources but no guaranteed bandwidth
- Performance issues for many Cloud apps
  - Network is the bottleneck
- Virtual Data Centers (VDCs)
  - Virtual machines, routers, switches and links

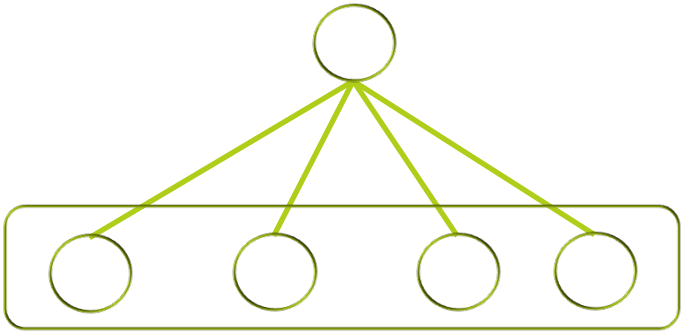


*M. F. Zhani, . Zhang, G. Simon, R. Boutaba. VDC Planner: Dynamic Migration-Aware Virtual Data Center Embedding for Clouds. IFIP/IEEE IM'13. Ghent (Belgium), May 27-31, 2013.

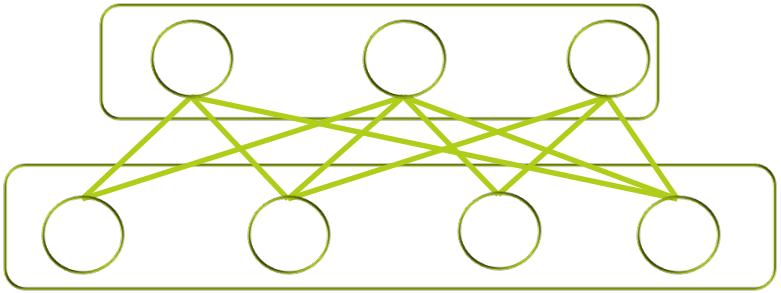
VDC Apps



Multi-tiered



Partition-Aggregate

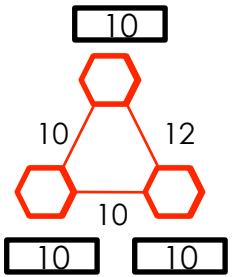


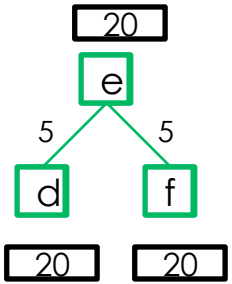
Bipartite

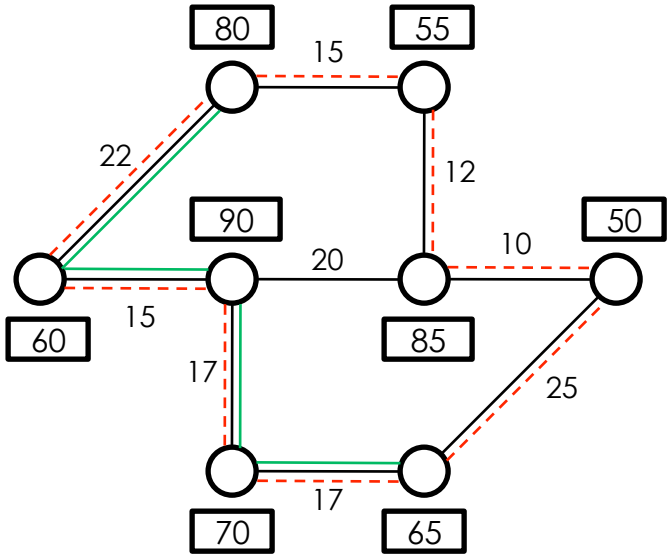
VN Embedding (VNE)

- VN is the basic entity of Network Virtualization
  - A collection of virtual nodes and virtual links forming a virtual topology
  - Mapped to physical nodes and links in the physical topology
- VN Embedding (VNE):
  - A virtual node is hosted on a particular physical node
    - Multiple virtual nodes can coexist
  - A virtual link spans over a physical path
    - Includes a portion of the underlying physical resources

VNE Example


10
10
12
10
10
10


20
e
5
5
d
f
20
20

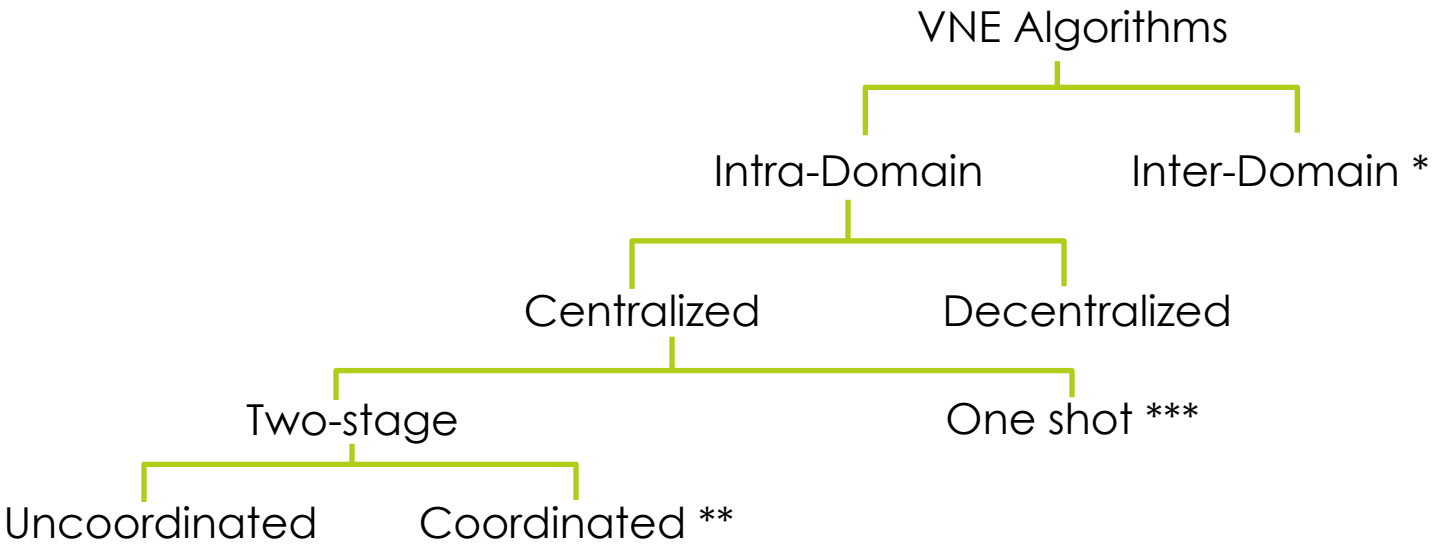

80
55
15
22
12
90
50
10
20
60
15
85
17
25
17
70
65

VNE Problem Formulation

Map Virtual nodes and virtual links of a VN onto physical infrastructure known as substrate network (SN)

- Objectives:
  - Maximize acceptance ratio/revenue
  - Minimize the scheduling delay
  - Maximize physical resource utilization
  - Minimize energy costs
- Achieve all/some of the objectives dynamically over-time, subject to a number of resource constraints

VNE Algorithms



* M. Chowdhury, F. Samuel, R. Boutaba. PolyViNE: Policy-based Virtual Network Embedding Across Multiple Domains. In the 2nd ACM SIGCOMM VISA Workshop, 2010.

** M. Chowdhury, M. R. Rahman and R. Boutaba. ViNEYard: Virtual Network Embedding Algorithms with Coordinated Node and Link Mapping. In IEEE/ACM Transactions on Networking, 2012.

*** R. Mijumbi, J. Serrat, J-L. Gorricho and R. Boutaba. A Path Generation Approach to Embedding of Virtual Networks. IEEE Transactions on Network and Service Management, 2015. .

What is missing?

- Existing solutions for VNE assume substrate network to be operational at all times
- However, substrate networks are error-prone
  - Single link failures commonly occur *
  - Node and multiple link failures are less frequent but not rare **
- Reliability aspect is missing from most VNE solutions

* Markopoulou, A *et al.* "Characterization of Failures in an Operational IP Backbone Network," *Networking, IEEE/ACM Transactions on* , vol.16, no.4, pp.749,762, Aug. 2008

** Phillipa Gill *et al.* Understanding network failures in data centers: measurement, analysis, and implications. In *Proceedings of the ACM SIGCOMM 2011*.

Reliability is Important !

- Failures: A major concern for service providers
  - A service outage can potentially incur high penalty in terms of revenue and customer satisfaction
  - Online businesses in North America lost more than $26.5 billion in revenue due to service downtime in 2010 *
  - Service downtime typically costs an organization $300,000 per hour **
- Reliability is a critical concern for SPs and InPs alike
  - Availability is a common QoS metric specified in SLAs (in number of 9s, e.g., 99.999)

* InformationWeek, "IT Downtime Costs" May 24, 2011. http://www.informationweek.com/it-downtime-costs-$265-billion-in-lost-revenue/d/d-id/1097919

** Gartner, "The Cost of Downtime" July 16, 2014. http://blogs.gartner.com/andrew-lerner/2014/07/16/the-cost-of-downtime/

Reliability in Virtualization

- Failures in Virtualized Environments
  - Failure affects multiple providers, cascading failures are common
  - Fault diagnosis is difficult because of the added layers of complexity and the lack of transparent metrics
  - Fault isolation is of paramount importance
  - Reactive procedures are nowhere near sufficient, e.g., re-provisioning light paths on the fly and meet 50ms recovery time (typical SLA in OTNs)
  - Proactive approaches are necessary to guarantee reliability
- Must be considered at VN creation time, *i.e.*, VN embedding

SVNE

- We coined the term and formulated the problem of Survivable Virtual Network Embedding (SVNE) in 2010 *
- Extended the VNE problem formulation to tackle SVNE
  - Assuming single link failure in the substrate network
- The VNE problem can be reduced to the *NP-hard multi-way separator* problem
- A VNE solution that can tolerate substrate resource failures is even harder!

InP pro-actively computes a set of possible backup detours for each substrate link using a path selection algorithm. When a substrate link fails, a reactive backup detour solution is invoked to reroute affected bandwidth along candidate backup detours.

* M. Rahman, I. Aib, and R. Boutaba. Survivable Virtual Network Embedding. Lecture Notes in Computer Science, 2010, Volume 6091, IFIP NETWORKING 2010, Pages 40-52. May 2010.

Recent Contributions

- Survivability with bandwidth guarantee for single link failure
  - SiMPLE
- Guaranteed network connectivity for multiple link failures
  - CoVine
- Dedicated protection for single node failure
  - DRONE
- Availability-aware embedding for virtual data centers
  - Venice

Outline

- Network Virtualization
  - Virtual Network
  - Virtual Network Embedding
- Reliability in Network Virtualization
  - Survivable Virtual Network Embedding (SVNE)
- Recent Contributions
  - SiMPLE
  - CoViNE
  - DRONE
  - VENICE
- Research Challenges/Directions

SiMPLE

- Survivability in Multi-Path Link Embedding
- A multi-path link embedding approach
  - Guarantees full bandwidth of a virtual link demand for single link failure
  - Provides higher salvaged bandwidth for multiple link failures
  - Requires much less backup bandwidth than existing approaches
- Contributions
  - SiMPLE – an embedding concept for guaranteeing survivability
  - SiMPLE-OP – an optimization model to trade-off survivability and overhead
  - SiMPLE-GR – a greedy heuristic for large problem instances

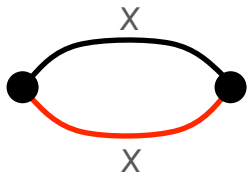
M.M.A. Khan, N. Shahriar, R. Ahmed, and R. Boutaba. SiMPLE: Survivability in multi-path link embedding. In the *11th International Conference Network and Service Management (CNSM), 2015,* Extended version in IEEE Transactions on Network and Service Management (TNSM), 2016.

SiMPLE Concept

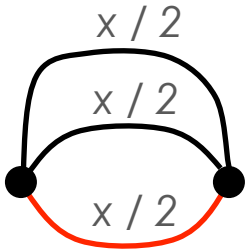
Virtual Link Demand = x

Primary path

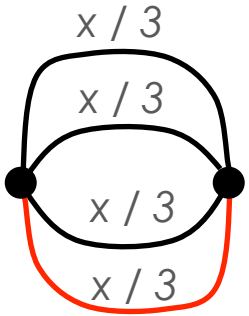
Backup path



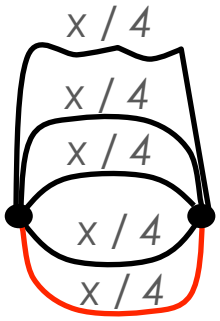
Base case (FBS)
BW Requirement = x + x



SiMPLE - Three splits
BW Requirement = x + x / 2
Backup BW Saving = 50%



SiMPLE - Four splits
BW Requirement = x + x / 3
Backup BW Saving = 67%



SiMPLE - Five splits
BW Requirement = x + x / 4
Backup BW Saving = 75%

SiMPLE-OP

- Formulated as an Integer Linear Programming (ILP) model
  - minimizes the physical resource consumption and path splitting overhead simultaneously
- Objective function

$$\textbf{\textit{minimize}}\ [\sum_{VLink}(SplitJoinCost + \sum_{Path}(SwitchCost + LinkCost + Delay))]$$

- Constraints
  - Substrate node and link capacities are not violated
  - Virtual node and link demands are satisfied
  - One virtual node must be mapped to one substrate node
  - Each substrate path for a virtual link must be link-disjoint

SiMPLE-GR

- Greedy algorithm for SiMPLE Link Embedding
  - Assume node embedding is done beforehand
  - For each virtual link in the VN
    - Computes the first *k* link-disjoint shortest paths for *k = 2, 3, 4, 5* using constrained Dijkstra's shortest path algorithm
    - Experiment shows more than 5 splits is not good
    - Returns the embedding yielding the lowest cost

Evaluation

- Compared approaches
  - SiMPLE-OP – ILP implementation using GLPK
  - SiMPLE-GR – implementation using C++
  - Full Backup Scheme (FBS) *
  - Shared Backup Scheme (SBS) **
- Evaluation focus
  - Fat tree topology for path diversity
  - Embedding performance in small scale topologies
  - Survivability analysis in large scale topologies
- Varying Parameters
  - Bandwidth demand
  - Failure rate

* M. R. Rahman *et al.*, Survivable Virtual Network Embedding, NETWORKING 2010, TNSM 2013
** T. Guo *et al.*, Shared Backup Network Provision for Virtual Network Embedding, ICC 2011

Key Results

- SiMPLE-GR performs very close to SiMPLE-OP
- SiMPLE achieves 50-100% more profit than FBS and SBS
- SiMPLE-GR reduces failure percentage by 30 – 50% over SBS
- SiMPLE-GR salvages 50 – 70% more bandwidth in case of multiple substrate failures compared to SBS and FBS
- SiMPLE uses 40 - 50% less backup bandwidth than FBS, and uses similar backup bandwidth to SBS

Outline

- Network Virtualization
  - Virtual Network
  - Virtual Network Embedding
- Reliability in Network Virtualization
  - Survivable Virtual Network Embedding
- Recent Contributions
  - SiMPLE
  - CoViNE
  - DRONE
  - VENICE
- Research Challenges/Directions

CoViNE

- Connectivity-aware Virtual Network Embedding*
- Guarantees connectivity in a VN
  - Requires no pre-allocated backup path and no path splitting
  - Survives multiple substrate link failures
  - SP reroutes traffic on the failed virtual links to alternate paths
  - Applicable to VNs carrying best-effort traffic
- Contributions
  - Conflicting set abstraction – to handle arbitrary number of failures
  - CoViNE-ILP - an ILP formulation for CoViNE
  - CoViNE-FAST - a fast and efficient heuristic for CoViNE

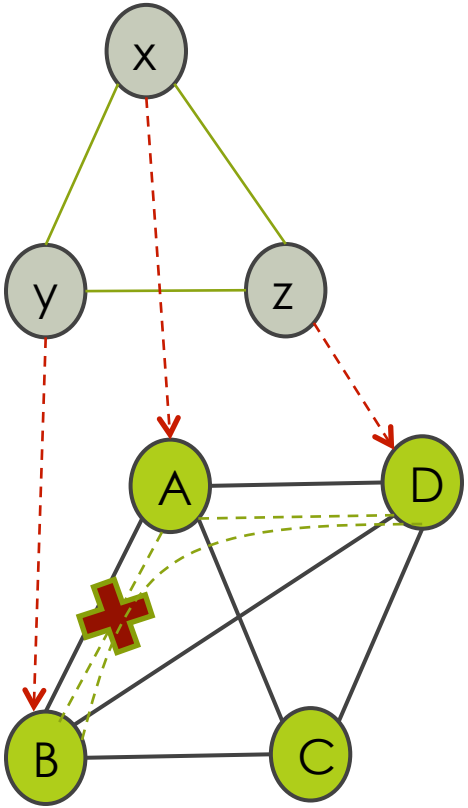
* N. Shahriar, R. Ahmed, S. R. Chowdhury, M. M. A. Khan, R. Boutaba, J. Mitra and F. Zeng. Connectivity-aware Virtual Network Embedding. To Appear in IFIP Networking, Vienna (Austria), May 17-19, 2016.

Problem Statement

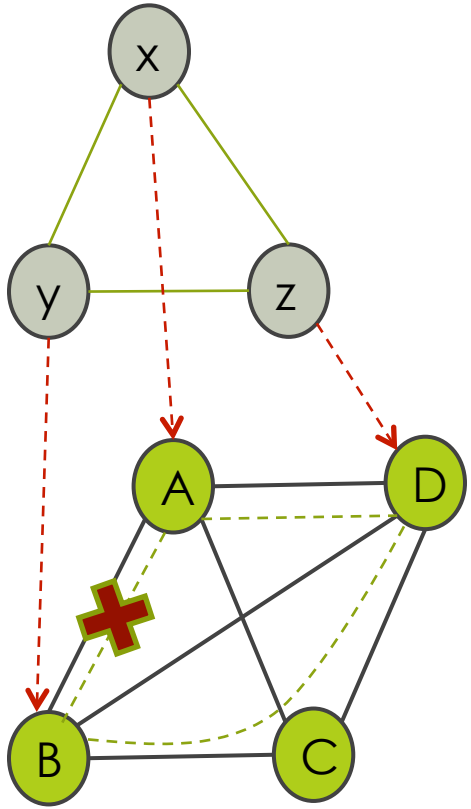
- Find an embedding that survives k link failures
  - Augment the VN topology to be k + 1 edge connected, i.e., k + 1 edge-disjoint virtual paths exist between each pair of virtual nodes*
  - Find which virtual links need to be embedded disjointedly so as to maintain k + 1 edge-disjoint paths between each pair of virtual nodes after the embedding
  - Embed the VN onto SN adhering to disjointedness requirement while minimizing the total cost of embedding

* Menger's theorem: https://en.wikipedia.org/wiki/Menger%27s_theorem

Single Link Failure Example

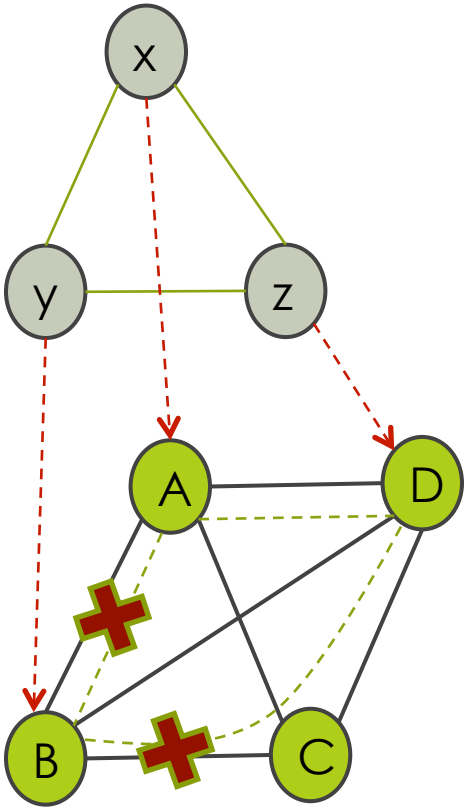


Un-survivable Embedding

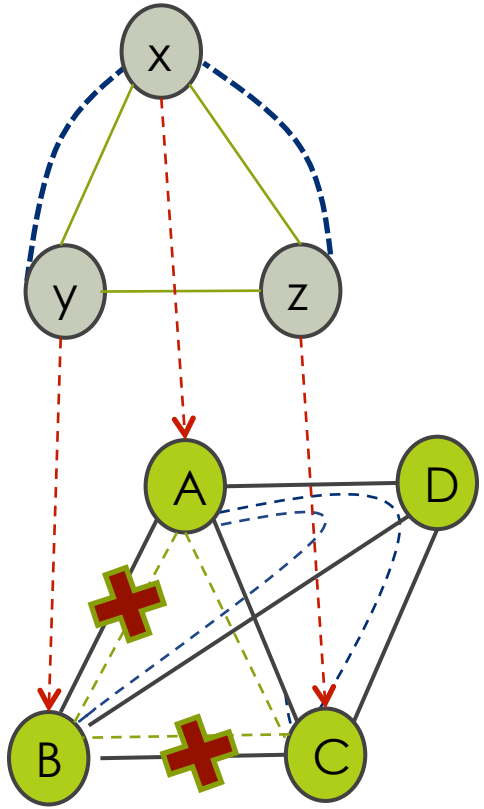


Survivable Embedding

Two-Link Failure Example



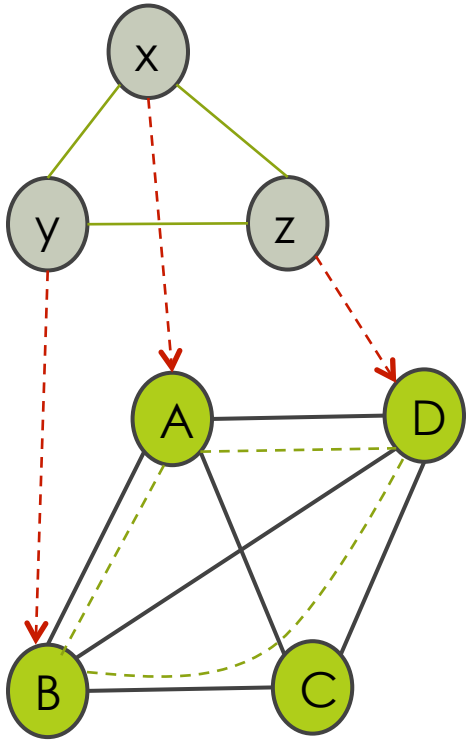
Un-survivable Embedding



Survivable Embedding

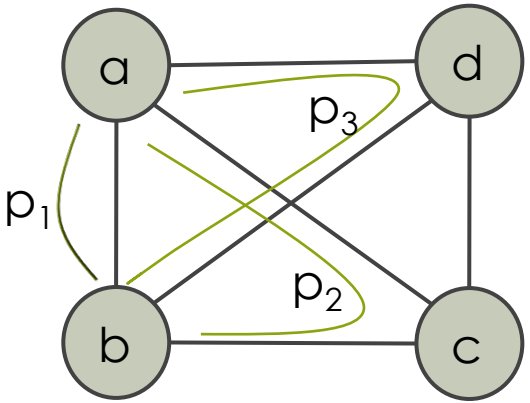
Conflicting Set Abstraction

- Two virtual links are conflicting if they must be embedded on disjoint paths
- Conflicting set is a function of the number of failures to survive
  - Set of links conflicting with a given link
- xy, yz, and zx are conflicting with each other for single failure survivability
  - Conflicting set of xy = {yz, zx}



Computing Conflicting Sets

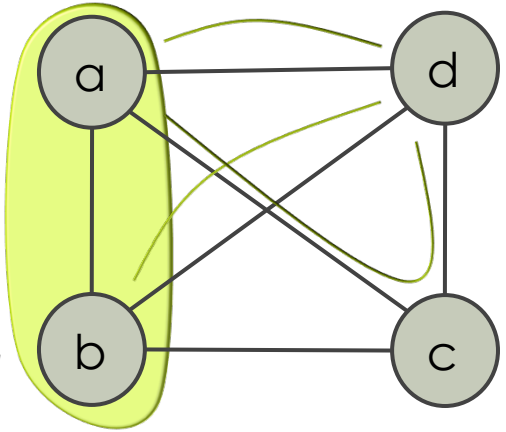
- Theorem : Computing optimal conflicting sets for all virtual links in a VN is *NP-complete*
  - Reduction from Minimum Vertex Coloring
- A heuristic algorithm to compute conflicting set of a link, *l*
  - For two endpoints of *l*, find k+1 edge-disjoint paths in the VN
  - *l* is conflicting with each link in other k paths
  - A link in an edge-disjoint path is conflicting with each link in all other paths
- $O(N^2)$ conflicting set computations!
  - Can be reduced to O(N)



Virtual Link ab
p_1 = {ab}
p_2 = {ac, bc}
p_3 = {ad, db}
Conflict set of ab = {ac, bc, ad, db}
Conflict set of ac = {ab, ad, db}

Computing Conflicting Sets (cont.)

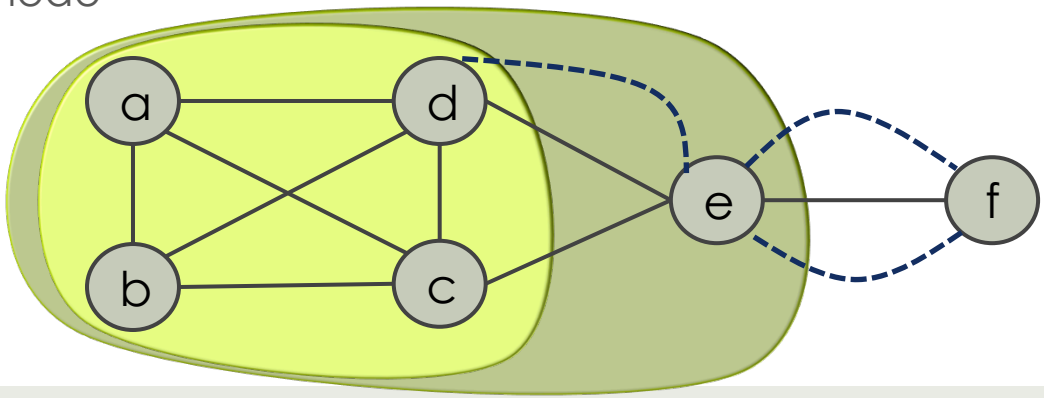
- Incremental k+1 edge-connected subgraph construction:
  - Start with a subgraph G of the VN containing a single virtual node (randomly chosen)
  - Repeat until all nodes are added to G
    - Select a node, *v* adjacent to a node in G
    - Find k+1 edge-disjoint paths from G to *v*
    - For all links in these paths, update conflicting sets
    - Add *v* to G
- Theorem: Incremental subgraph construction yields smaller conflicting sets
- Only considers virtual links in MST



3 edge-connected subgraph (a, b) and Virtual Link ad
p_1 = {ad}
p_2 = {bd}
p_3 = {ac, cd}
Conflict set of ad = {…, bd, ac, cd}
No need to compute for bd

VN Augmentation

- Augmentation of VNs with less than k+1 edge connectivity
  - Add max(0, k+1-m) parallel virtual links between a k+1 subgraph, *G* and a virtual node, *v* not in *G*
    - *m* is the number of edge-disjoint paths from *G* to *v*
  - Does not change pairwise connectivity of the VN nodes
  - Theorem: number of added links is fixed irrespective of the starting virtual node



CoVine Embedding: Formulation

- Objective: Minimize the total bandwidth cost

$$\text{minimize} \sum_{l' \in E'} \sum_{l \in P_{l'}} c(l) \times b$$

  - $c(l)$: cost of unit bandwidth on physical link l
  - B: bandwidth demand of virtual link l'
  - $P_{l'}$: physical path on which l' is embedded
  - E' : set of virtual links

- Subject to disjointedness constraints per conflicting sets in addition to other VN embedding constraints

CoViNE Embedding: Solutions

- CoViNE-ILP - Optimal solution
  - Extends Multi-Commodity Unsplittable Flow problem with disjointedness constraint
- CoViNE-FAST – Fast heuristic for large scale topologies
  - Constrained Shortest Path First algorithm
  - Link mapping satisfying disjointedness constraints and minimizing cost determines the node mapping

Evaluation

- Compared approaches
  - CoViNE-ILP : ILP implementation using CPLEX
  - CoViNE-FAST : C++ implementation
  - Cutset-ILP : Optimal solution for single failure scenario *
  - ViNE-ILP : Optimal solution for VN embedding **
- Embedding evaluation parameters:
  - Network size : 50 - 1000
  - Link to node ratio : 1.2 - 4
- Survivability analysis:
  - 3 traffic classes with different priorities
  - Single and two-link failure scenarios

* E. Modiano et al., "Survivable lightpath routing: a new approach to the design of wdm-based networks," IEEE JSAC, 2002.
** Y. Zhu et al., "Algorithms for assigning substrate network resources to virtual network components," in IEEE INFOCOM, 2006.

Key Results

- CoViNE-FAST allocates ~10%, ~15%, and 18% more bandwidth than CoViNE-ILP, Cutset-ILP, and ViNE-ILP, respectively
  - 2 to 3 orders of magnitude faster than ILP counterparts
  - Scalable to thousand-node topologies, not possible by ILP
- Two-Link link failure survivability requires ~30% more bandwidth than that for single failures
  - Embedding cost of parallel virtual links dominates in sparse VNs
  - Satisfying disjointness constraints dominates otherwise
- Restores ~100% bandwidth for the highest priority traffic
  - Penalizes lower priority traffic
  - Restored bandwidth by ViNE-ILP is worst due to partitioning

Outline

- Network Virtualization
  - Virtual Network
  - Virtual Network Embedding
- Reliability in Network Virtualization
  - Survivable Virtual Network Embedding
- Recent Contributions
  - SiMPLE
  - CoViNE
  - DRONE
  - VENICE
- Research Challenges/Directions

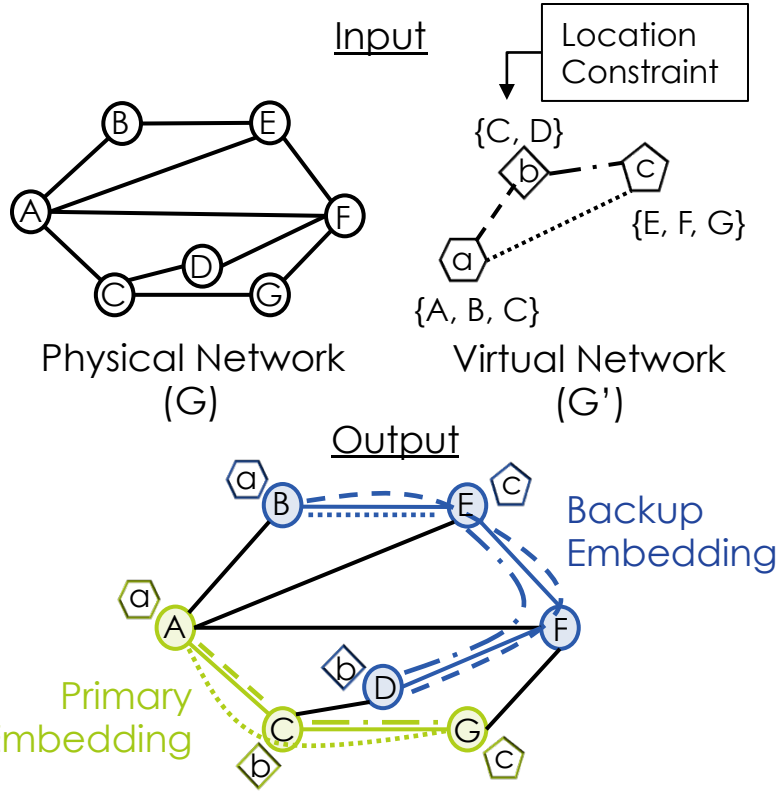
DRONE

- Dedicated Protection for Virtual Network Embedding*
  - A suit of solutions to the 1 + 1–Protected Virtual Network Embedding (1 + 1-ProViNE) problem
- 1 + 1 – ProViNE
  - Provides dedicated backup for each virtual node and link
    - Required to meet 50ms VN recovery time, typical in optical transport networks
  - Guaranteed VN survivability during single physical node failure
- Contributions
  - OPT-DRONE: ILP based optimal solution to 1+1-ProViNE
  - FAST-DRONE: Fast heuristic for 1+1-ProViNE

* S.R. Chowdhury, R. Ahmed, MMA. Khan, N. Shahriar, R. Boutaba, J. Mitra, and F. Zeng. Protecting Virtual Networks with DRONE. To Appear in IEEE/IFIP NOMS, 2016

1+1-ProViNE Problem Statement

- Given
  - A physical network G = (V, E)
  - A virtual network G' = (V', E')
  - Location constraints for embedding virtual nodes
- Find two disjoint embeddings of the nodes and links of G' on G such that
  - Nodes of G' have two disjoint embeddings on G
  - Links of G' have two disjoint embedding paths on G
  - The total bandwidth cost is minimized



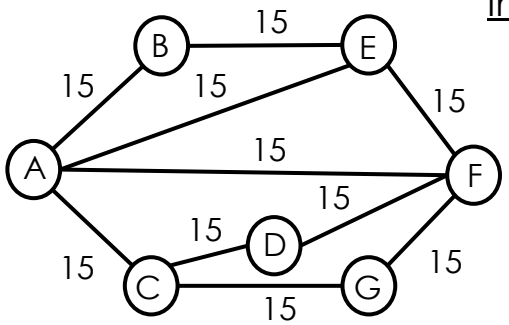
OPT-DRONE

- Formulated as an Integer Linear Program (ILP)
  - Objective: minimize the physical bandwidth allocation cost for the primary and backup embeddings.
- Constraints
  - No over commitment of physical resources
  - A single virtual link cannot be mapped to multiple physical paths, i.e., unsplittable virtual links
  - Virtual node embedding should satisfy location constraint
  - The primary embedding is node and link disjoint from the backup embedding

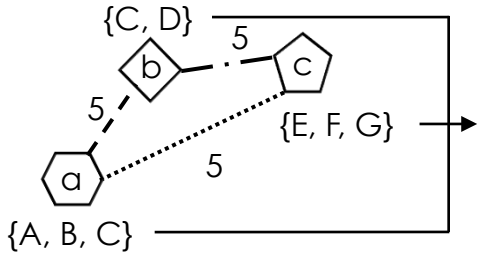
FAST-DRONE

- Solution Approach
  - Reformulate problem as special case of graph partitioning without loosing original semantic

Input



A physical network graph, G = (V, E)

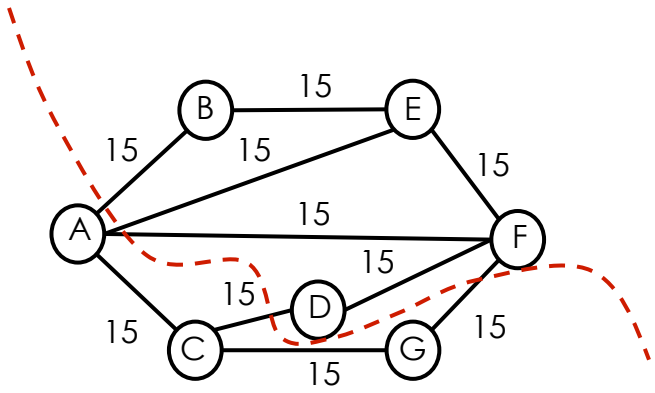


A virtual network graph G'=(V', E')

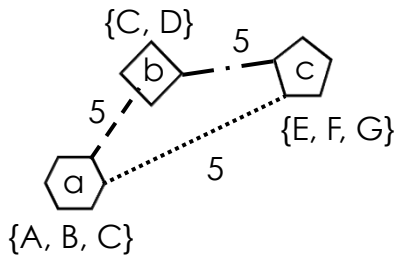
A set of location constraints, $L = \{L_{U'} \mid L_{U'} \subseteq V, \forall U' \in V'\}$

FAST-DRONE (cont)

- Partition G into two node-disjoint partitions P and Q for the primary and backup embeddings respectively



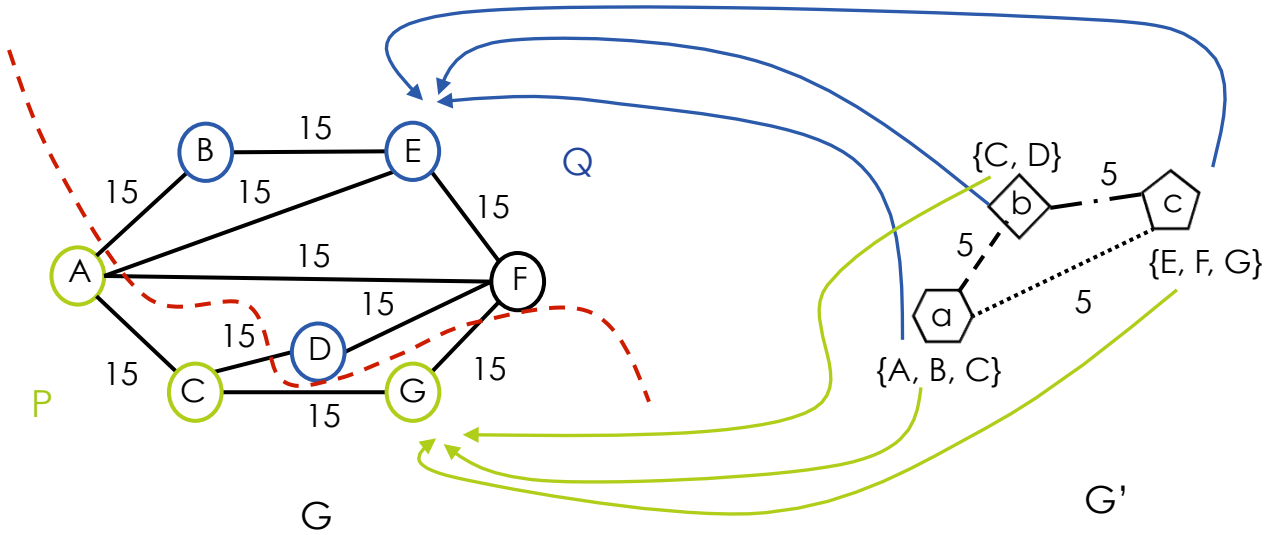
G



G'

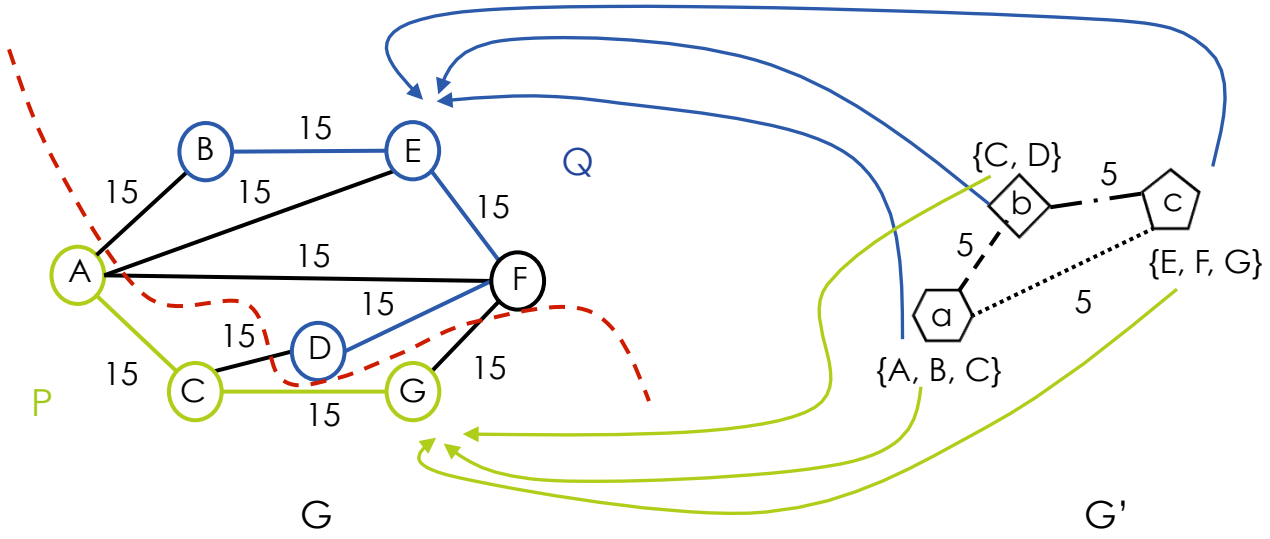
FAST-DRONE (cont)

- P satisfies at least one location constraint from each set L_U,
- Q satisfies at least one location constraint from each set L_U,



FAST-DRONE (cont)

- The nodes satisfying the location constraints of L_U, in P and Q are connected



FAST-DRONE (cont)

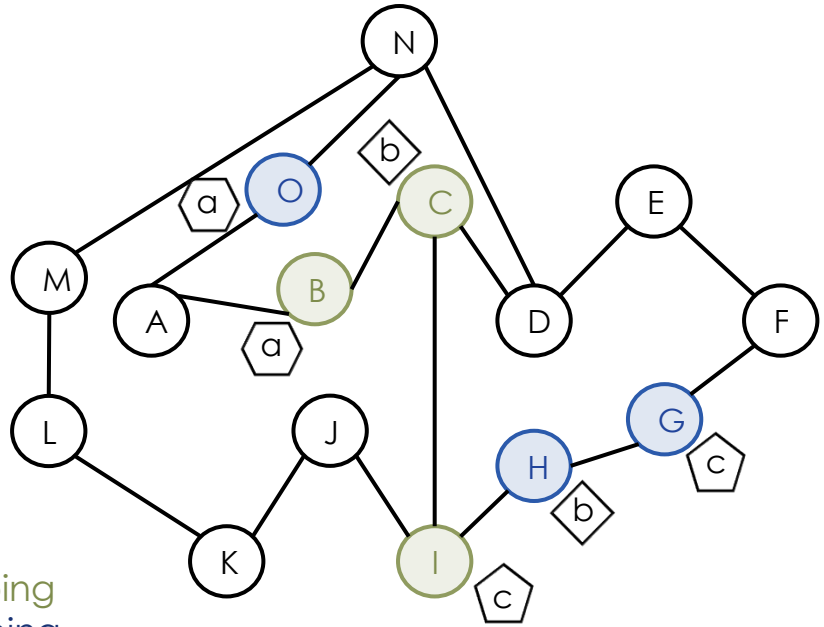
- Once we have the partitions:
  - Solution to multi-commodity unsplittable flow with unknown source and destination in each partition gives node and link embeddings.
  - The partition that yields the minimum cost is the optimal solution.
  - NP-Hard!
- FAST-DRONE heuristic
  - Finds a near-optimal partitioning of the physical network
  - Embeds the VN onto the two partitions

FAST-DRONE Heuristic

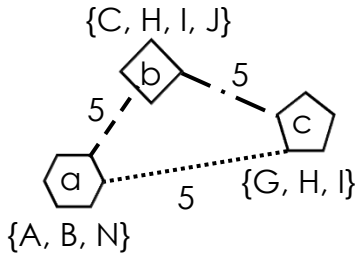
- Three-phase algorithm for 1+1-ProViNE
  - Node Mapping Phase: Map the virtual nodes in the order of most constrained to least constrained virtual node while minimizing probability of infeasible partitioning.
  - Physical Network Partitioning Phase: Partition the physical network into a primary and backup set based on the virtual node mapping.
  - Link Mapping Phase: For each virtual link, map the virtual link on the shortest path between the mapped nodes of the link's endpoints in both primary and backup partition.

FAST-DRONE in Action

- Phase-1: Node Mapping Phase



Physical Network - G



Virtual Network - G'

Output:

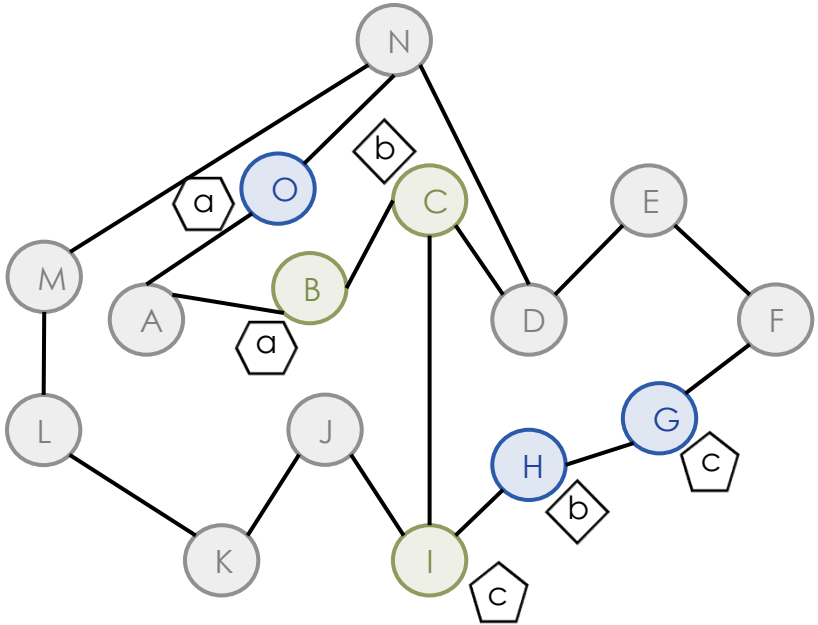
Primary Mapping

Backup Mapping

FAST-DRONE in Action (cont)

- Phase-2: Partitioning Phase

Input:
Seed Primary Partition
Seed Backup Partition
Un-partitioned Nodes

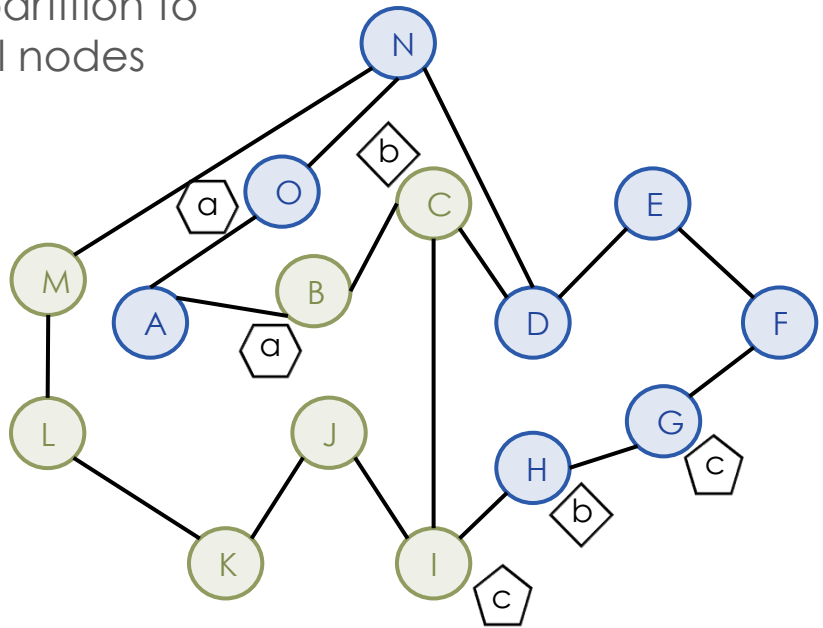


Physical Network - G

FAST-DRONE in Action (cont)

- Phase-2: Partitioning Phase (cont)
  - Grow each seed partition to include all physical nodes

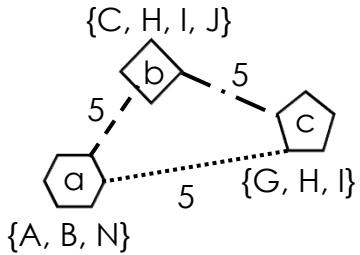
Output:
Primary Partition
Backup Partition



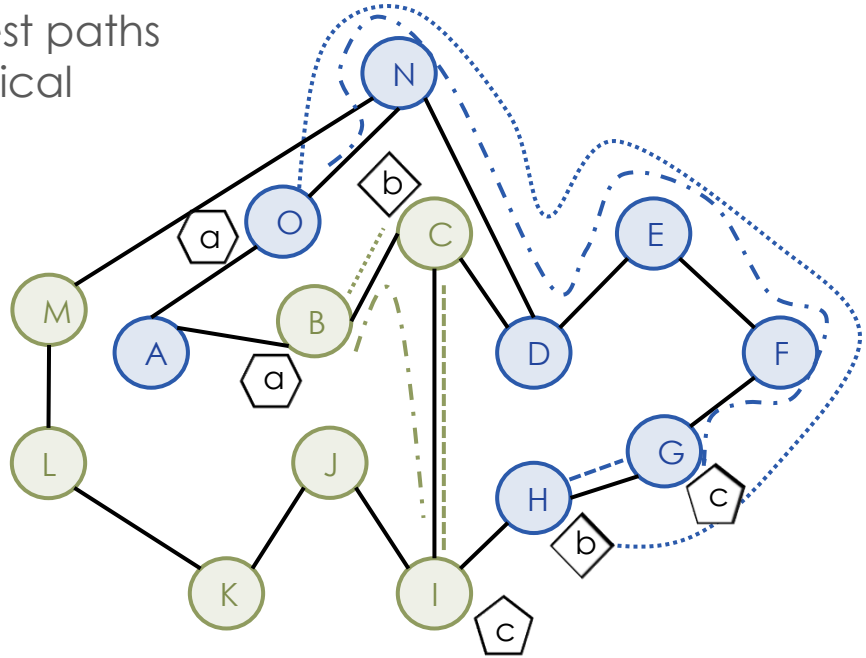
Physical Network - G

FAST-DRONE in Action (cont)

- Phase-3: Virtual Link Mapping Phase
  - Map virtual links to the shortest paths between their mapped physical nodes



Virtual Network – G'



Physical Network G

Evaluation

- FAST-DRONE compared with OPT-DRONE and state-of-the-art solution*
- Physical Network
  - Synthetic topologies
  - Node count between 50 -150
  - Average node degree between 2.4 – 4.4
- Virtual Networks with <=16 virtual nodes

* Z. Ye, A.N. Patel, P. N. Ji, C. Qiao. "Survivable Virtual Infrastructure Mapping With Dedicated Protection in Transport Software Defined Networks." Journal of Optical Communications and Networking 7(2): A183-A189, 2015.

Key Results

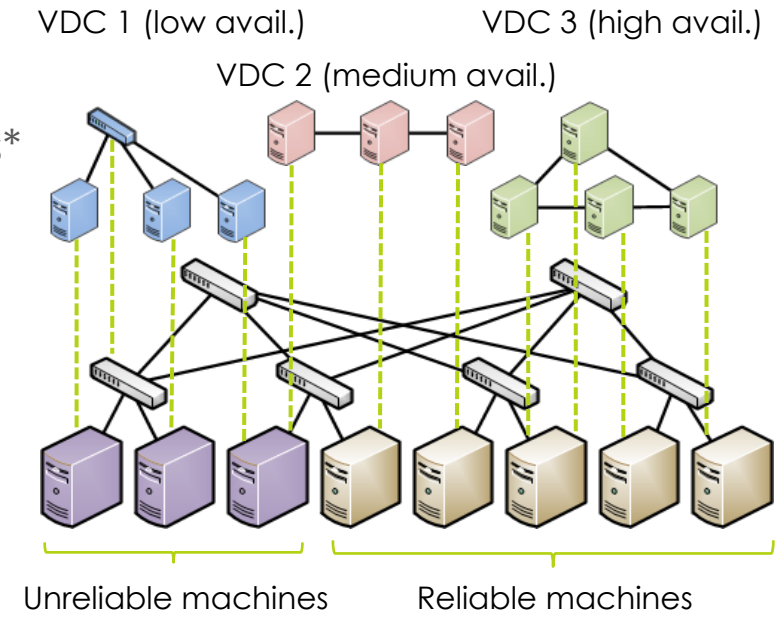
- FAST-DRONE allocates ~15% extra bandwidth on average compared to the optimal solution, i.e., OPT-DRONE, while executing 2 – 3 orders of magnitude faster
- FAST-DRONE performs better for physical networks with average degrees <= 3.6 (typical for ISP Networks)
- FAST-DRONE allocates 17.5% less bandwidth and accepts 4x more VNs on average compared to state-of-the-art solution

Outline

- Network Virtualization
  - Virtual Network
  - Virtual Network Embedding
- Reliability in Network Virtualization
  - Survivable Virtual Network Embedding
- Sample Research Contributions
  - SiMPLE
  - CoVine
  - DRONE
  - VENICE
- Research Challenges/Directions

Venice

- VDCs have heterogeneous availability requirements*
- Resources have heterogeneous availability characteristics
- Place VDCs with high availability requirement on reliable machines



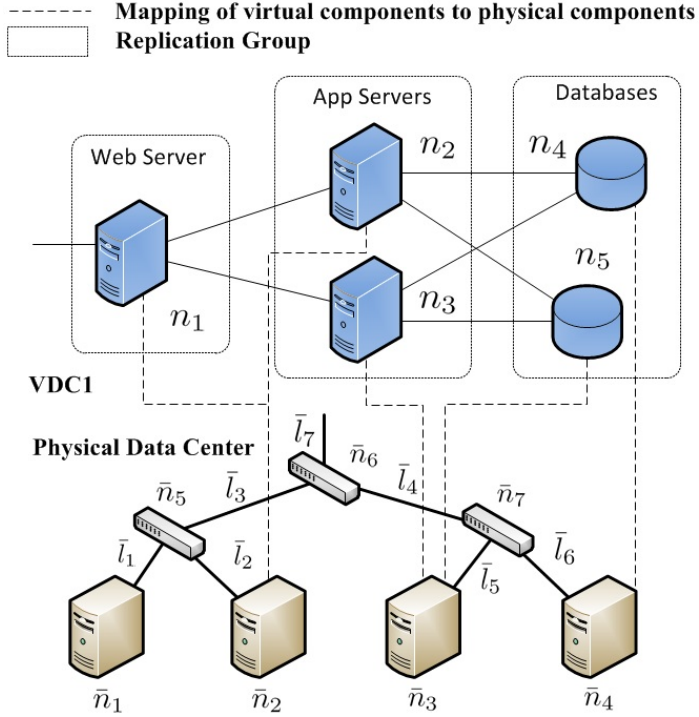
* Q. Zhang, M. F. Zhani, M. Jabri, R. Boutaba. Venice: Reliable Virtual Data Center Embedding in Clouds. IEEE INFOCOM'14, Toronto, ON (Canada), April 27 - May 2, 2014.

Computing VDC Availability

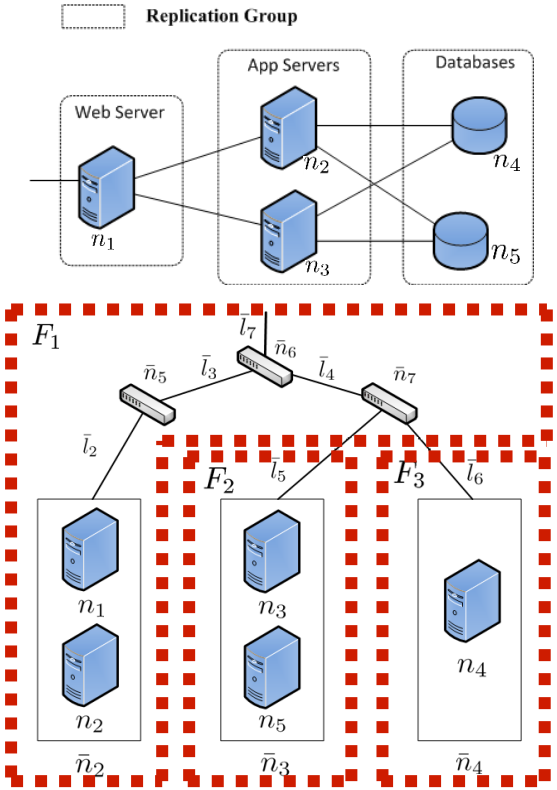
- Example of 3-tier application
- Availability of device *j*:

$$A_j = \frac{MTBF_j}{MTBF_j + MTTR_j}$$

- How to compute the availability of this VDC?



Computing VDC Availability (cont)



- Identify all possible failure scenarios S^k and compute the availability

Case 1: F1 unavailable,

$$A_{F_1} = 0$$

Prob. of occurrence: $P(F_i) = 1 - \prod_{i \in F_1} A_i$

Case 2: F1 available, F2 unavailable

$$A_{F_1} = \prod_{i \in F_2} A_i$$

Prob. of occurrence: $P(F_2) = (\prod_{i \in F_1} A_i)(1 - \prod_{i \in F_2} A_i)$

Case 3: F1 available, F2 available

$$A_{F_1} = 1$$

Prob. of occurrence: $P(F_2) = \prod_{i \in F_1 \cup F_2} A_i$

Using conditional probability, the availability of VDC_1 can be computed as:

$$A_{VDC_1} = \sum_{i=1}^{3} P(F_i) A_{F_i}$$

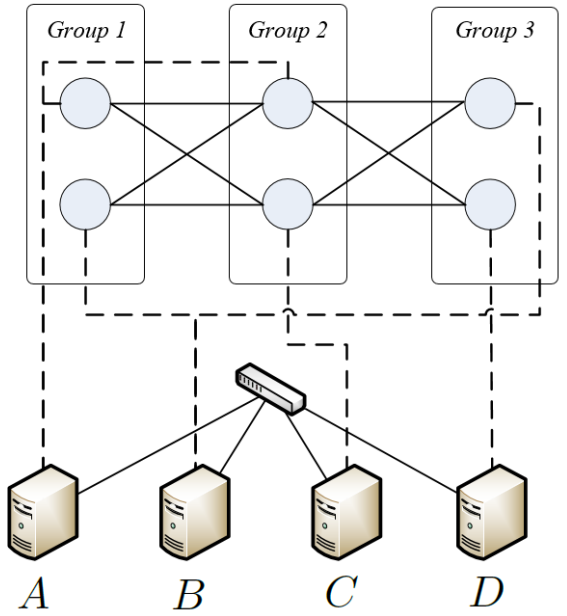
Computing VDC Availability (cont)

Theorem: VDC availability cannot be computed in polynomial time in the general case

Proof: Reduction from the counting monotone 2-Satisfiability problem

… Need to consider an exponential number of scenarios in the worst case!

$$f(A,B,C,D) = (A \vee B) \wedge (A \vee C) \wedge (B \vee D)$$



Computing VDC Availability (cont)

- Observation: it is unlikely to see large simultaneous failures
  - Given 3 nodes, each with availability > 95%, the probability of seeing all 3 nodes fail simultaneously is at most $(1-0.95)^3<0.00013$
- A fast heuristic:
  - Compute availability using scenarios S^k that involve at most 3 simultaneous failures
- Fast heuristic provides a lower bound on VDC availability

Problem Formulation

- Objective function:

$$\min C_E + C_M + C_A$$

- Where

$$C_E = \sum_{\bar{n} \in \bar{N}} y_{\bar{n}} p_{\bar{n}} \qquad \text{(Resource cost)}$$

$$C_M = \sum_{i \in I} \sum_{n \in N^i} \sum_{\bar{n} \in \bar{N}} \gamma_n x^i_{n\bar{n}} g^i_{n\bar{n}} \qquad \text{(Migration cost)}$$

$$\mathcal{C}_A = \sum_{i \in I} (1 - A_i)\pi_i + \sum_{\bar{n} \in \bar{N}} F_{\bar{n}} C^{restore}_{\bar{n}} + \sum_{\bar{l} \in \bar{L}} F_{\bar{l}} C^{restore}_{\bar{l}} \qquad \text{(Failure cost)}$$

Greedy Scheduling Algorithm

- For each received VDC request
  - Initial embedding: embed one node from each replication group.
  - Repeat
    - For each remaining component compute a score as the availability improvement - resource cost
    - Embed the component with the highest score
  - Until the VDC availability is achieved or all nodes are embedded
  - Embed the remaining components greedily based solely on resource cost

Experiments

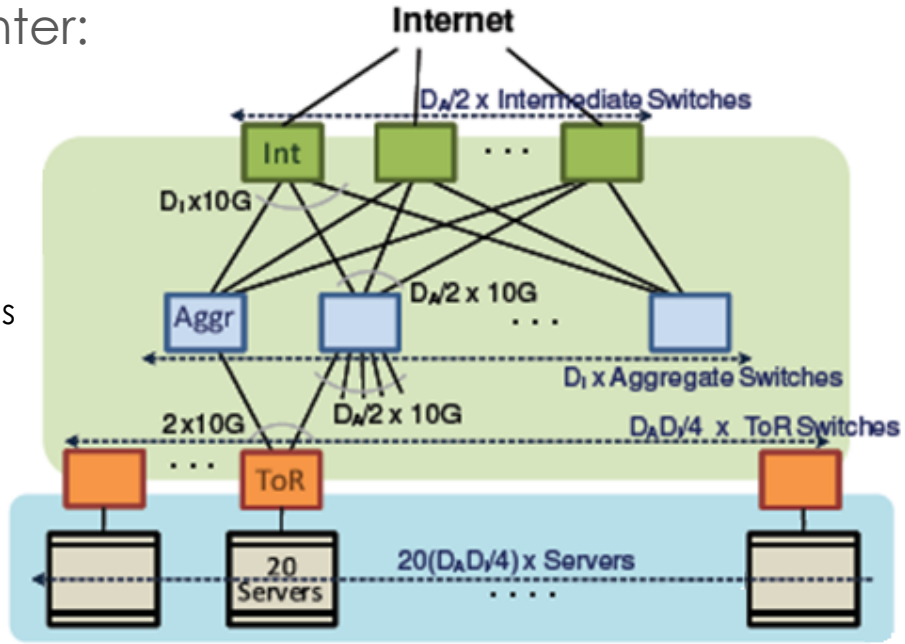
- Physical data center:

4 core switches

4 aggregation switches

4 top-of-rack switches

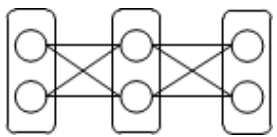
400 physical machines
(8 Cores, 8GB, 100 GB disk).



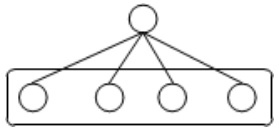
VL2 Topology

Experiment Setup

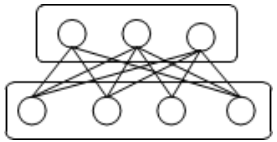
- VDC request formats
  - From 1 to 10 VMs per group
  - Different availability requirements
- VDC Planner used as a baseline for comparison



(a) Multi-tiered



(b) Partition-Aggregate

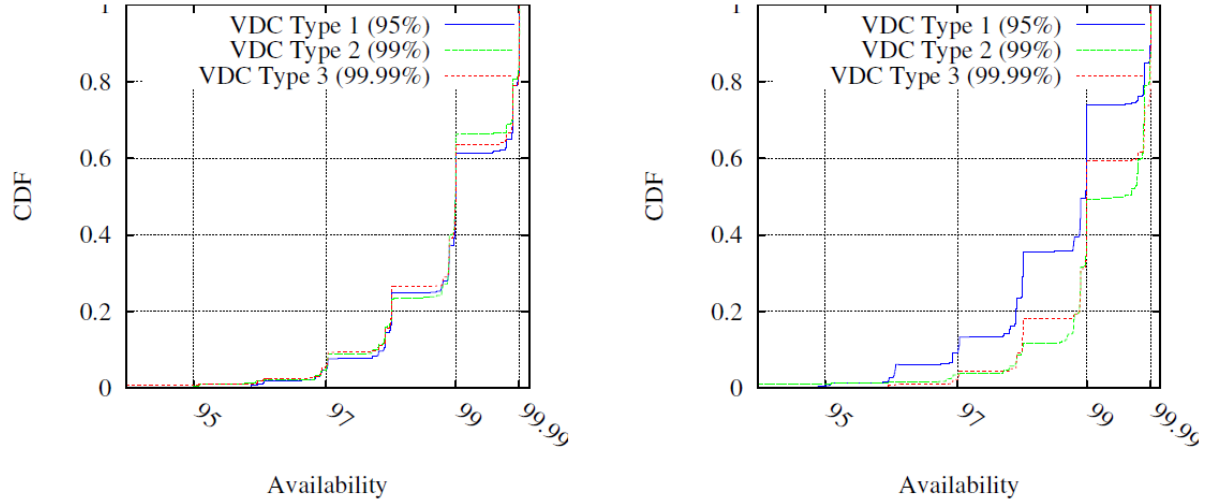


(c) Bipartite

TABLE I: VDC Availability requirements

VDC Type	Minimum Required Availability (%)	Acceptable daily downtime
1	95.00	1h:12mn
2	99.00	14mn:2s
3	99.99	08.64s

Results: Availability

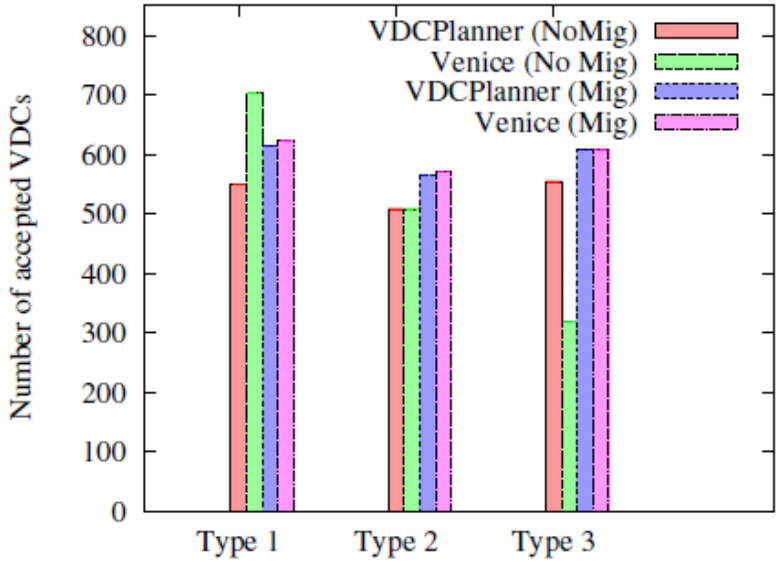


(a) VDC Planner (using migration)

(b) Venice (using migration)

- Venice increases the number of VDCs satisfying availability requirements by up to 35%

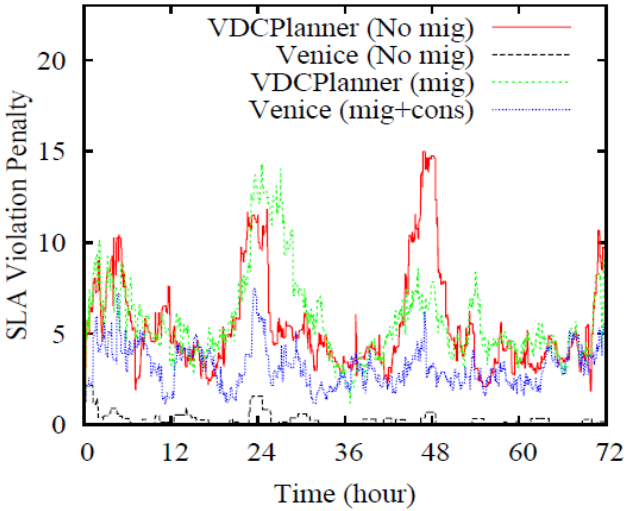
Results: Acceptance Ratio



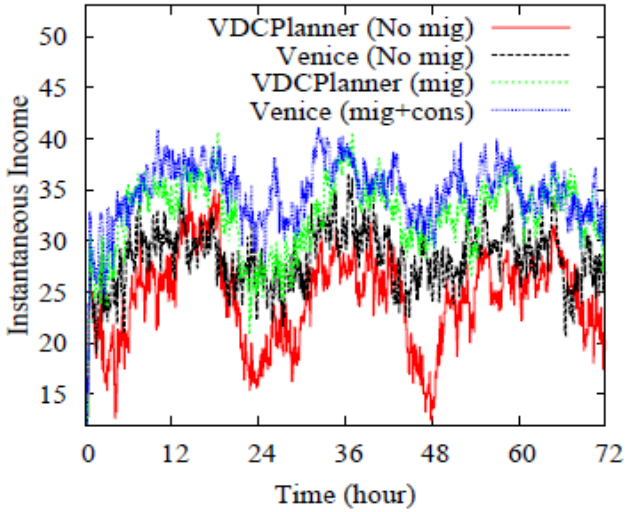
Number of accepted VDCs

- With migration, the number of accepted VDCs is comparable to that of VDC Planner

Results: Revenue



SLA Violation Cost



Instantaneous Income Rate

- Venice achieves 15% increase in revenue compared to VDC Planner

Summary

- We have discussed different reliability schemes
  - survivability with bandwidth guarantee for single link failure
  - connectivity with best-effort restored bandwidth for multiple failure scenarios
  - dedicated protection for single node and link failure
  - availability-aware virtual data center embedding
- Each incurs cost of provisioning additional resources
  - Customers may want to trade-off cost with reliability
- How to achieve different levels of reliability for different parts of a heterogeneous virtual network?
  - Can empower a wide variety of Service Level agreements

Research Challenges

- Virtualization: Benefit at what Cost ?
  - Increased layers of complexity; high demand variance; Infrastructure cost shedding; Overlay underlay tussle
- Virtual Solutions Can Generate Real Crisis
  - A lot to learn from the financial market meltdowns
    - De-regulation and the inevitable decay of overwatch; The double-fake scheme (derivatives, financial engineering); Self-interest vs. collective well-being
- New questions to ponder on
  - Risk – Lack of methodology to assess risk
  - Accountability – No clear division of responsibility
  - System-wise stability – not guaranteed in most ecosystems

Research Directions

- Building Management Frameworks for reliable virtualized environments
  - Interface between multiple management paradigms
  - Draw clear line between the management responsibilities of the InPs and the SPs
  - Design cross layer reliability mechanisms without losing transparency and isolation
  - Well-grounded and transparent metric system that can relate virtual quantities and qualities to physical counterparts.

Acknowledgement

Md Mashrur Alam Khan

Nashid Shahriar

Reaz Ahmed

Shihabur Rahman Chowdhury

Qi Zhang

Mohamed Faten Zhani

Thank you!