Connectivity-aware Virtual Network Embedding

Nashid Shahriar, Reaz Ahmed, Shihabur R. Chowdhury, Md Mashrur Alam Khan, Raouf Boutaba

Jeebak Mitra, Feng Zeng

UNIVERSITY OF WATERLOO
FACULTY OF MATHEMATICS
David R. Cheriton School of Computer Science

HUAWEI

Outline

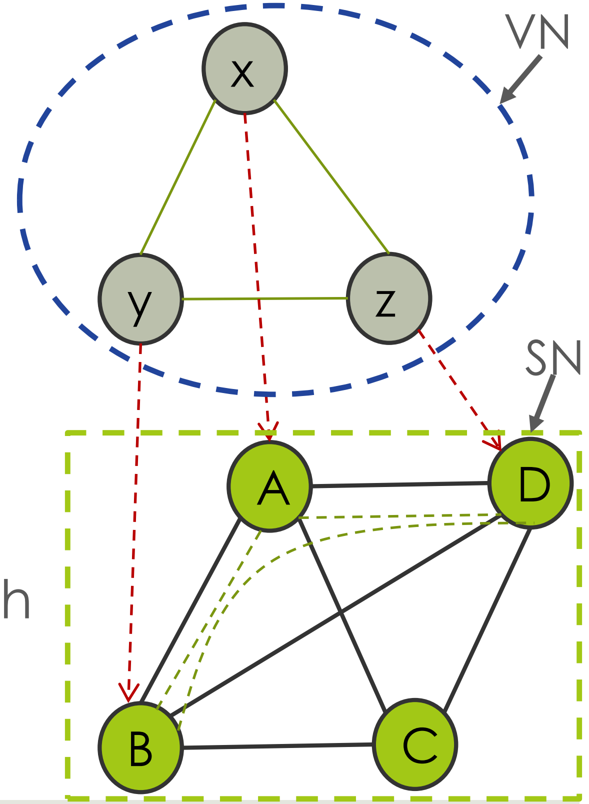
- Survivability in Virtual Network Embedding (VNE)
- Connectivity-aware Virtual Network Embedding (CoViNE)
  - State of the art
  - Solution approaches
    - CoViNE-ILP
    - CoViNE-Fast
  - Evaluation
- Summary and future work

Outline

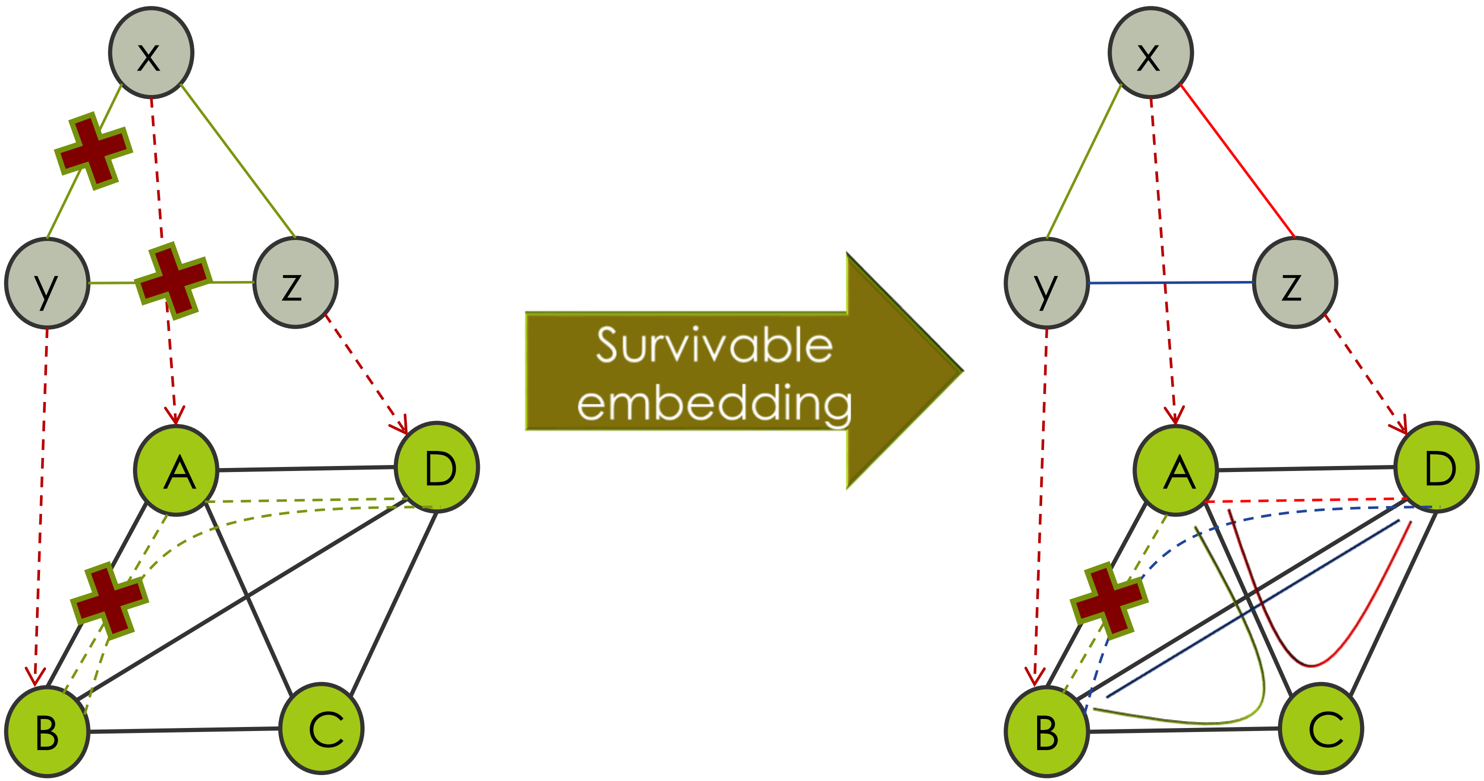
- **Survivability in Virtual Network Embedding (VNE)**
- Connectivity-aware Virtual Network Embedding (CoViNE)
  - State of the art
  - Solution approaches
    - CoViNE-ILP
    - CoViNE-Fast
  - Evaluation
- Summary and future work

Virtual Network Embedding

- A virtual network (VN) is a collection of virtual nodes and virtual links
  - Embedded on a substrate network (SN)
- A virtual node is hosted on a substrate node
  - Multiple virtual nodes can coexist
- A virtual link spans over a substrate path
  - Link capacities are not exceeded

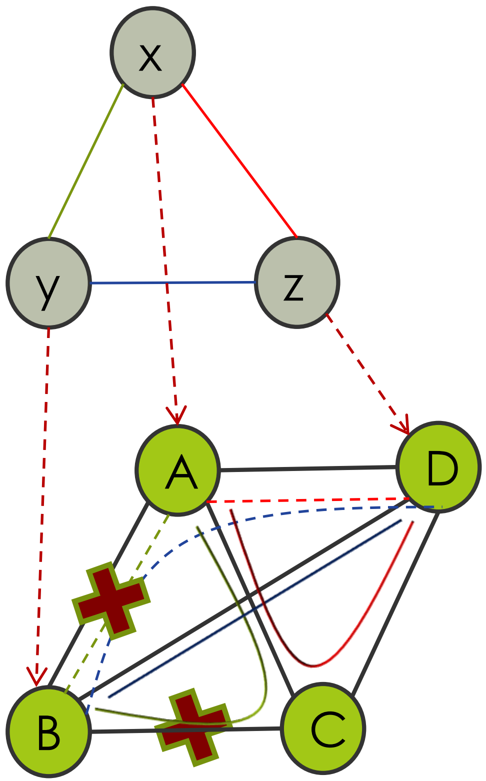


Survivability in VNE (SVNE)



Survivability in VNE (SVNE)

- Limitations of traditional SVNE
  - Requires pre-allocated backup path disjoint from the primary path
    - Wastage of expensive resources
  - Sharing of backup path possible
    - Sacrifices level of survivability
  - Cannot survive arbitrary failure scenarios
    - Multiple substrate link failures

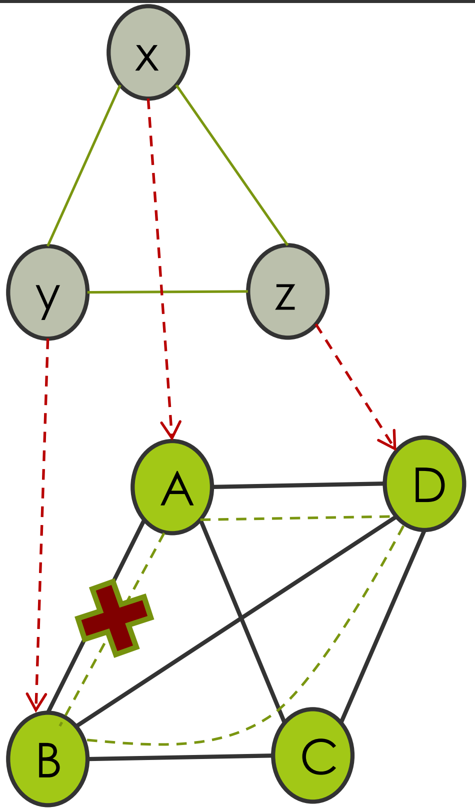


Outline

- Survivability in Virtual Network Embedding (VNE)
- **Connectivity-aware Virtual Network Embedding (CoViNE)**
  - State of the art
  - Solution approaches
    - CoViNE-ILP
    - CoViNE-Fast
  - Evaluation
- Summary and future work

Connectivity-aware VNE (CoViNE)

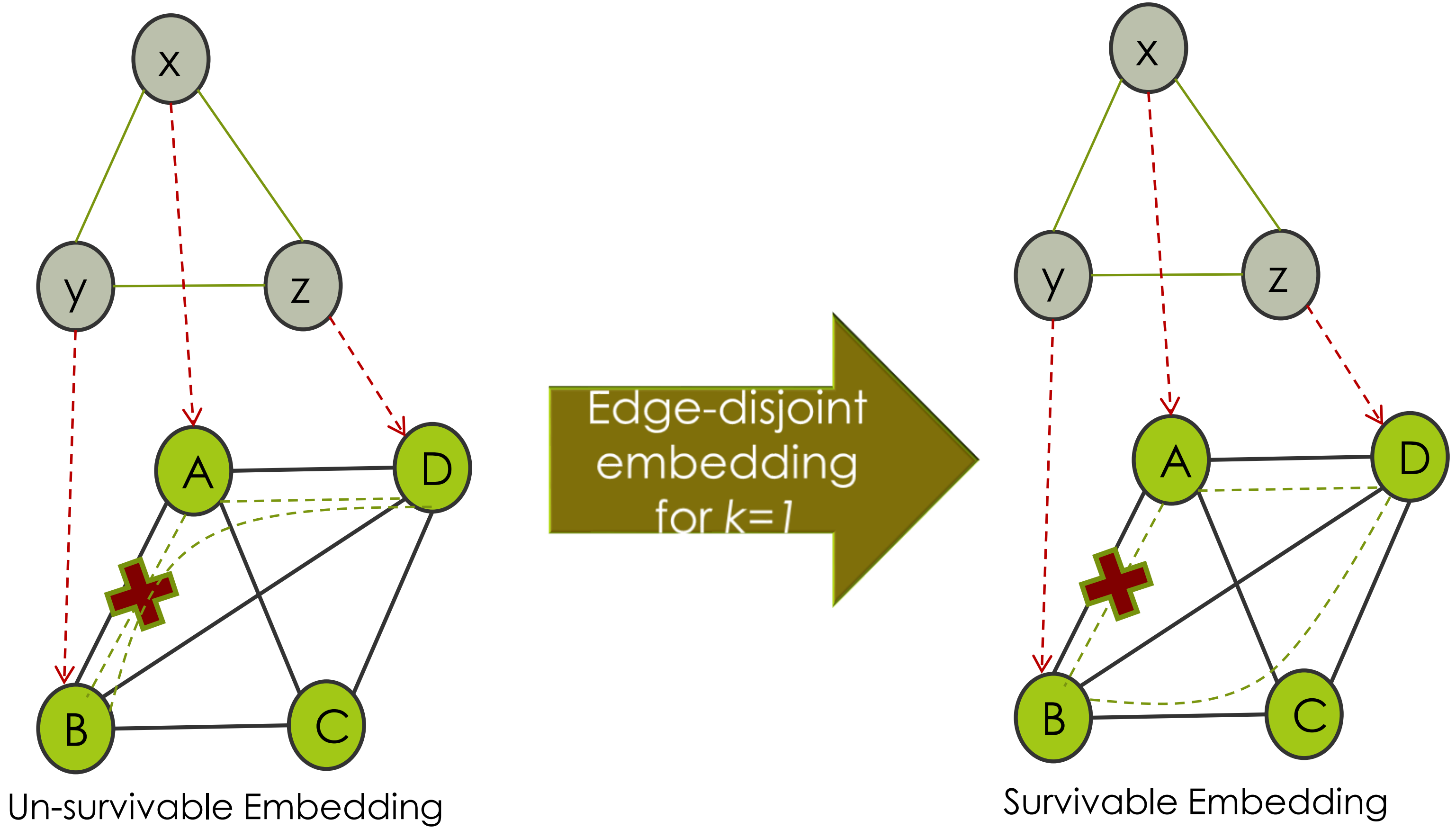
- A weaker form of survivability
  - Guarantees connectivity of a VN
    - Less backup resource needed
  - Computes alternate path upon failure
    - Traffic is rerouted based on priority thanks to SDN controller
  - Suitable for carrying best-effort traffic
    - Tolerates small amount of delay



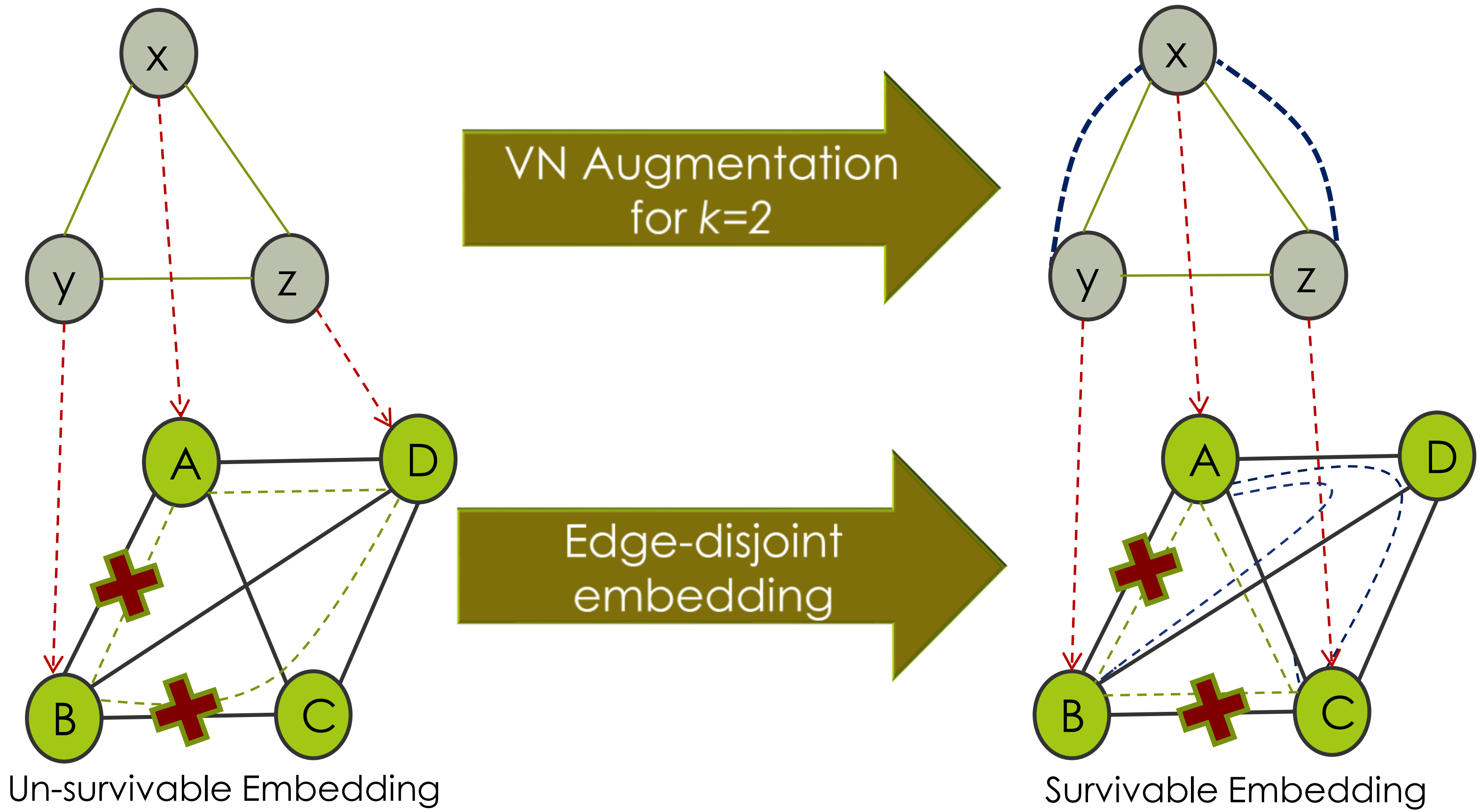
CoViNE Key Question

How to resource efficiently embed a VN while ensuring connectivity under multiple *(k)* substrate link failures?

CoViNE challenges



CoViNE Challenges



Problem Statement

- Decomposed sub-problems
  - Augment the VN to make it *k + 1* edge connected
    - *k + 1* edge-disjoint virtual paths exist between each pair of virtual nodes*
  - Identify sets of virtual links to be embedded disjointedly
    - Ensures *k + 1* edge-disjoint paths between each pair of virtual nodes in the embedding
  - Embed the augmented VN onto SN
    - Adheres to disjointedness constraints while minimizing total cost of embedding

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

Outline

- Survivability in Virtual Network Embedding (VNE)
- Connectivity-aware Virtual Network Embedding (CoViNE)
  - **State of the art**
  - Solution approaches
    - CoViNE-ILP
    - CoViNE-Fast
  - Evaluation
- Summary and future work

State of the Art

- Not studied in network virtualization context
- A special case in IP-over-WDM network literature for IP connectivity
  - Do not consider node embedding

Approach	Limitation
Cut-set based approach *	Only applicable to *k=1*, not scalable
Survivable Mapping Algorithm by Ring Trimming**	Fails to deal with arbitrary topology and multiple failures
Logical topology augmentation for guaranteed survivability***	Generates large number of disjointedness constraints

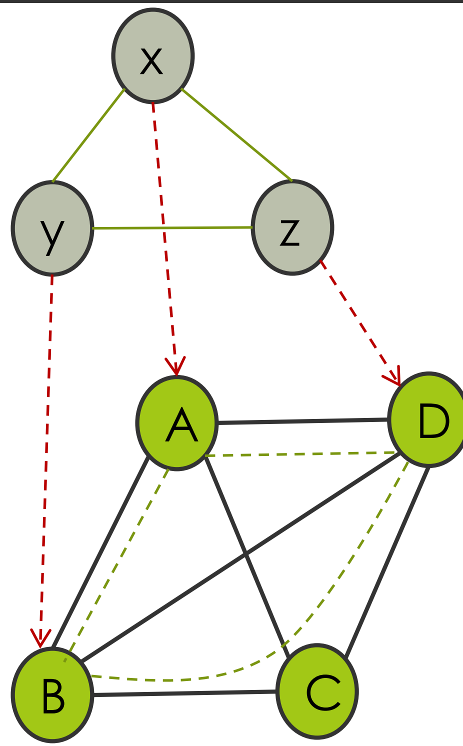
* E. Modiano et al., "Survivable lightpath routing: a new approach to the design of wdm-based networks," IEEE JSAC, 2002.
** M. Kurant et al., "Survivable mapping algorithm by ring trimming (smart) for large ip-over-wdm networks," in BroadNets, 2004.
*** K. Thulasiraman et al., "Logical topology augmentation for guaranteed survivability under multiple failures in ip-over-wdm optical networks," Optical Switching and Networking, vol. 7, no. 4, pp. 206–214, 2010.

Outline

- Survivability in Virtual Network Embedding (VNE)
- Connectivity-aware Virtual Network Embedding (CoViNE)
  - State of the art
  - **Solution approaches**
    - CoViNE-ILP
    - CoViNE-Fast
  - Evaluation
- Summary and future work

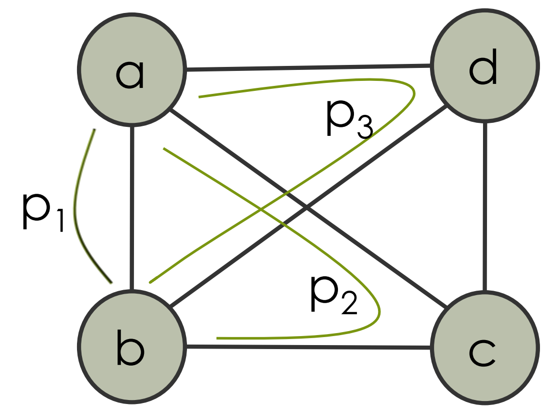
Conflicting Set Abstraction

- Two virtual links are conflicting if they must be embedded on disjoint paths
- Conflicting set is a function of *k*
  - Set of links conflicting with a given link
- xy, *yz*, and zx are conflicting with each other for *k=1*
  - Conflicting set of *xy* = *{yz, zx}*



Computing Conflicting Sets

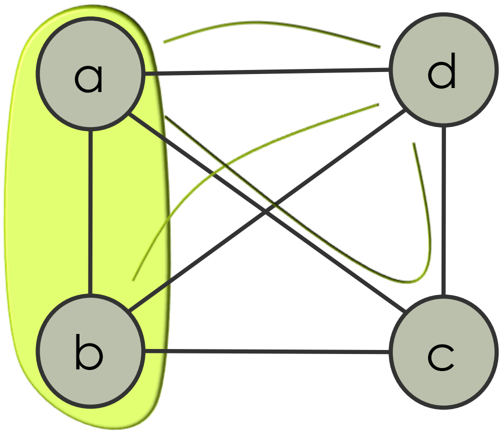
- Computing the optimal conflicting sets for all virtual links in a VN is *NP-complete*
  - Reduction from Minimum Vertex Coloring
- A heuristic algorithm to compute conflicting set of a link, *ab*
  - For two endpoints of *ab*, find *k+1* edge-disjoint paths in the VN
  - *ab* is conflicting with each 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.)

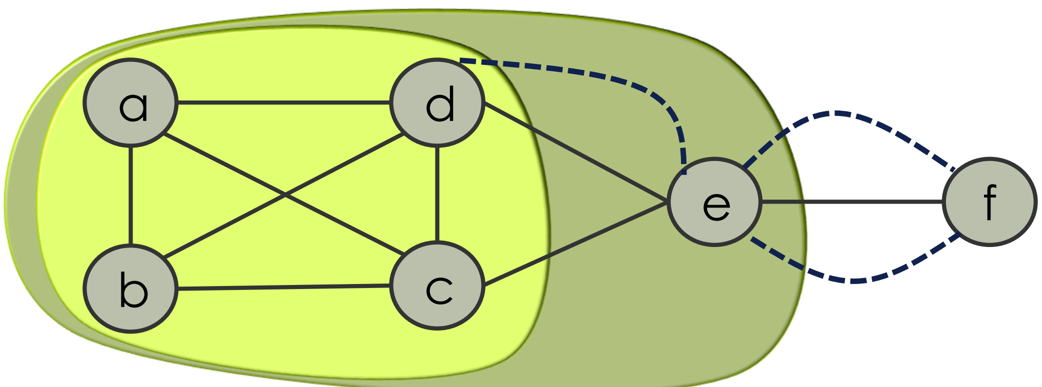
- Incremental *k+1* edge-connected sub-graph construction
  - Start with a sub-graph G of the VN containing a randomly chosen node
  - Repeat until all nodes are added to G
    - Select a node, *v* adjacent to a node in G
    - Find *k+1* edge-disjoint paths from G to *v*
    - For all links in these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



3 edge-connected sub-graph *(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* edge-connected sub-graph, *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 patterns of the virtual nodes



CoViNE-ILP

- An integer linear programming formulation for embedding a VN
  - Minimize total bandwidth cost

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

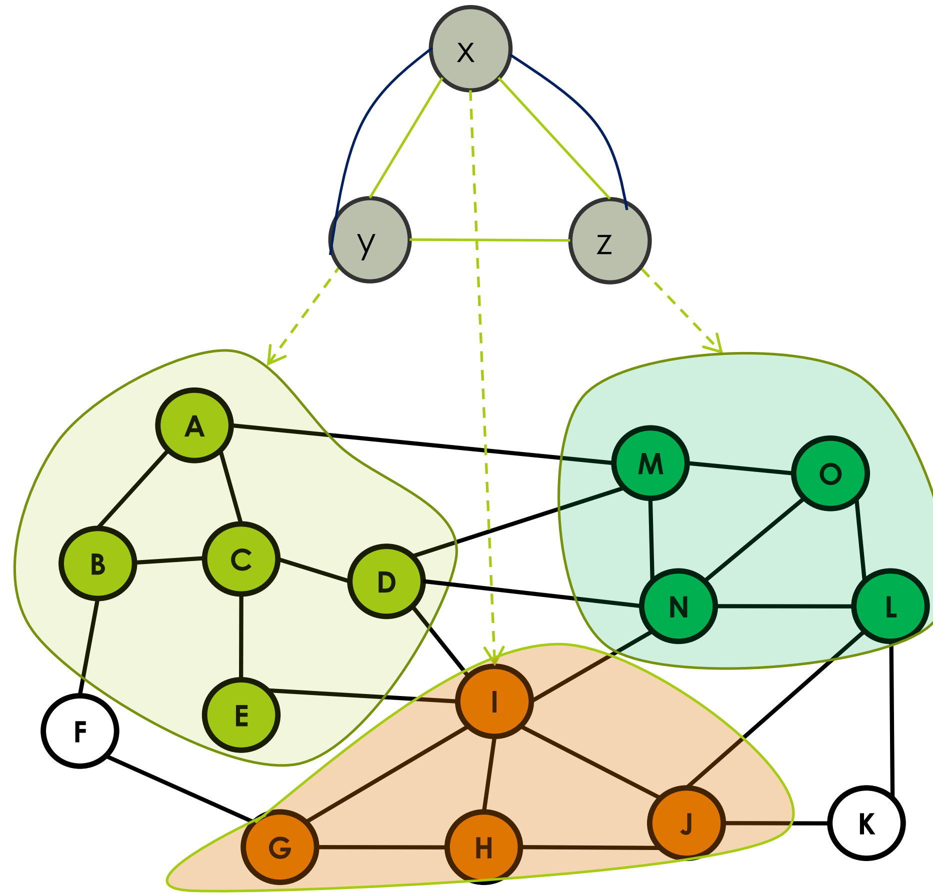
    - *c(l)*: cost of unit bandwidth on substrate link *l*
    - *b*: bandwidth demand of virtual link *l'*
    - $P_{l'}$: substrate path on which *l'* is embedded
    - *E'* : set of virtual links
  - Constraints
    - Node mapping satisfies location constraints
    - A virtual link is only mapped to a single substrate path
    - Link mapping adheres to disjointedness constraints
    - No over commitment of substrate resource capacity

CoViNE-Fast

- Fast and scalable heuristic algorithm
  - Node mapping
    - Minimizes total cost of mapping incident virtual links
    - Adheres to given location constraints of virtual nodes
    - Maps virtual nodes to substrate nodes in a greedy manner
  - Link mapping
    - Minimizes cost of mapped substrate path
    - Satisfies disjointedness constraints
    - Based on the constrained minimum cost path first algorithm
      - Modified version of Dijkstra's shortest path algorithm
  - Node and link mapping in a coordinated manner

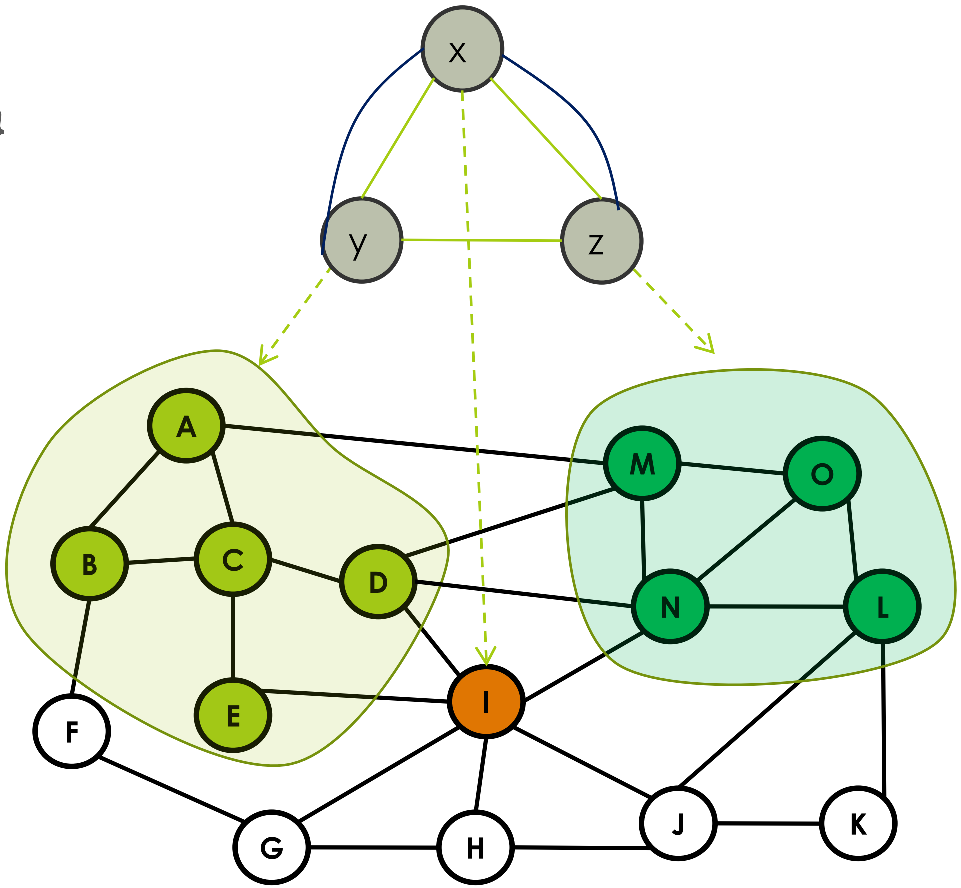
CoViNE-Fast in action

- Iteration for x
  - Location *(x) = {H, I, J, G}*
  - Compute minimum cost substrate paths from H
    - *P(xy) ={HI-ID}*
    - *P(xy)' ={HJ-JI-IN-ND}*
    - *P(xz) ={HG-GI-IN}*
    - *P(xz)' ={HJ-JL-LN}*
  - Compute similarly for *I, J, G*
  - Let, *I* yields minimum cost
    - Map x to *I*



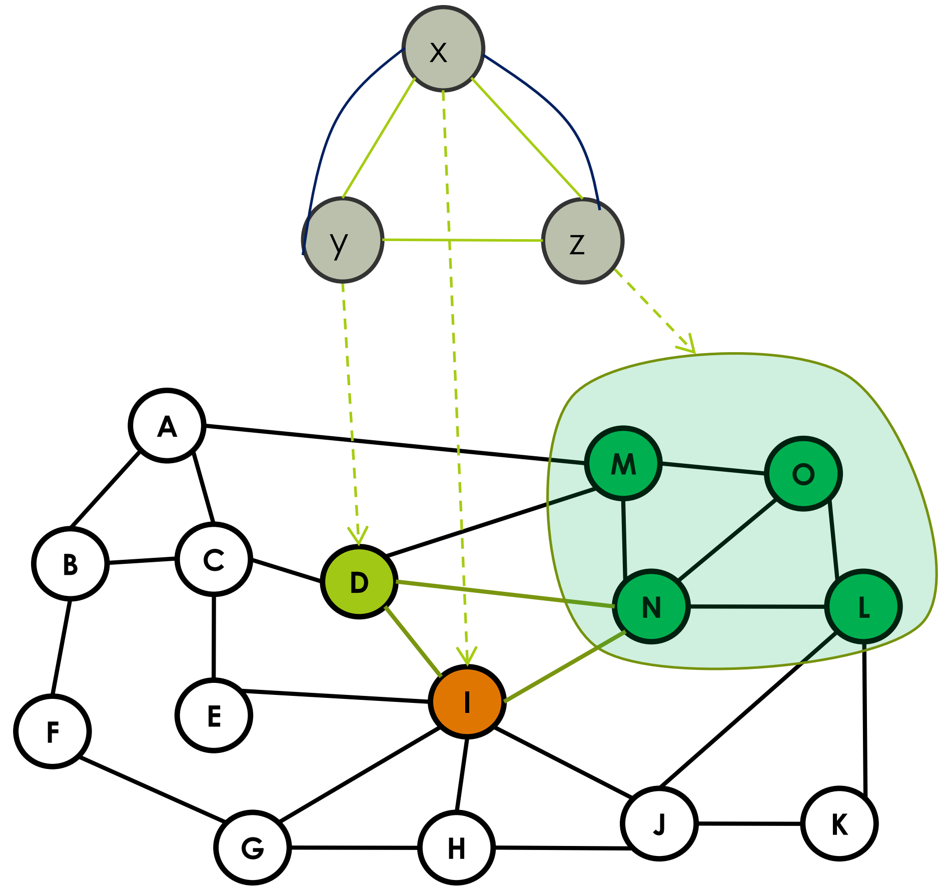
CoViNE-Fast in action

- Iteration for *y*
  - Location *(y)* = *{C, D, E, A, B}*
  - Compute minimum cost substrate paths from C
    - *P(xy) ={CD-DI}*
    - *P(xy)'={CE-EI}*
    - *P(yz) ={CA-AM}*
  - If *D* yields minimum cost
    - Map *y* to *D*
    - Map xy and *(xy)'*
      - *M(xy) = {ID}*
      - *M(xy)' = {IN, ND}*

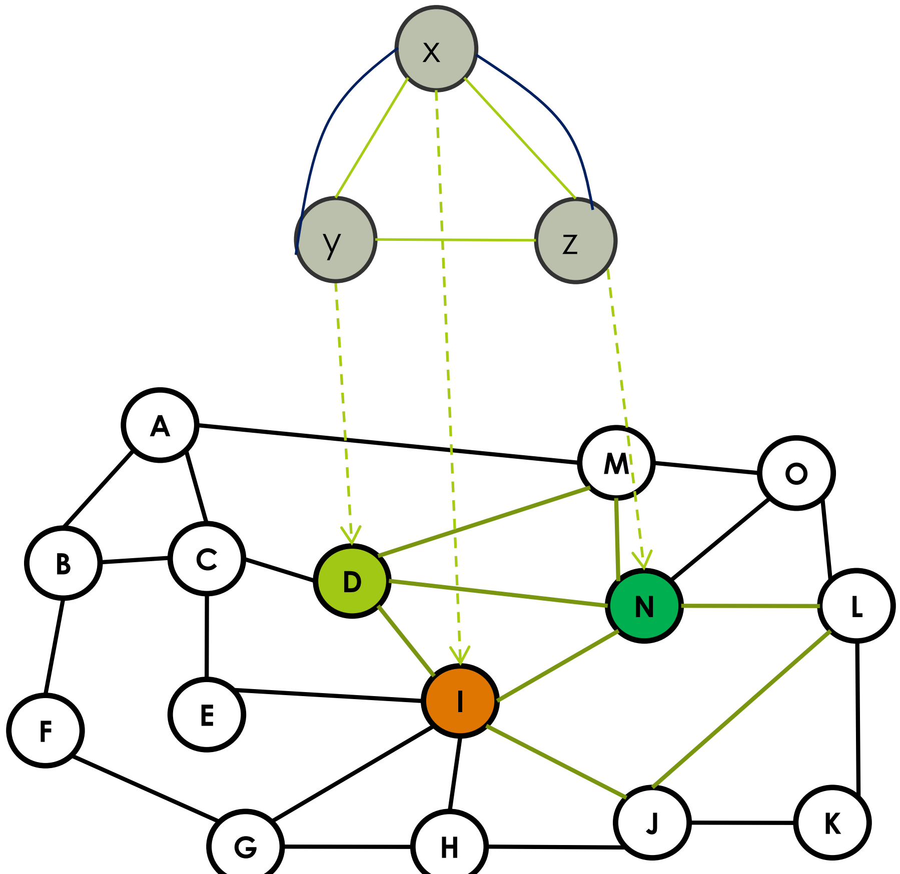


CoViNE-Fast in action

- Iteration for z
  - Location *(z)* = *{N, M, L, O}*
  - Compute minimum cost substrate paths from *N*
    - *P(xz) ={NL-LJ-JI}*
    - *P(xz)'={IN}*
    - *P(yz) ={NM-MD}*
  - If *N* yields minimum cost
    - Map z to *N*
    - Map *yz, xz,* and *(xz)'*
      - *M(yz) = {NM-MD}*
      - *M(xz) = {NL-LJ-JI}*
      - *M(xz)' = {IN}*



CoViNE-Fast embedding



Outline

- Survivability in Virtual Network Embedding (VNE)
- Connectivity-aware Virtual Network Embedding (CoViNE)
  - State of the art
  - Solution approaches
    - CoViNE-ILP
    - CoViNE-Fast
  - **Evaluation**
- Summary and future work

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 disjointedness constraints dominates otherwise
- Restores ~100% bandwidth for the highest priority traffic
  - Penalizes lower priority traffic
  - Restored bandwidth by ViNE-ILP is worst due to VN partitioning

Outline

- Survivability in Virtual Network Embedding (VNE)
- Connectivity-aware Virtual Network Embedding (CoViNE)
  - State of the art
  - Solution approaches
    - CoViNE-ILP
    - CoViNE-Fast
  - Evaluation
- **Summary and future work**

Summary

- Generic solutions to CoViNE for multiple substrate link failure
  - Conflicting set abstracts the number of failures
    - A heuristic algorithm to compute conflicting sets
  - ILP formulation for CoViNE embedding
  - A heuristic algorithm to reduce computational complexity
- Compared to the optimal, the heuristic algorithm
  - Allocates ~15% extra resources on average
  - Runs 2 to 3 orders of magnitude faster
  - Scales to thousands of node topologies

Future Work

- Extend current solutions to consider
  - Spare bandwidth allocation to guarantee bandwidth
  - Node throughput constraints for better utilization
  - Substrate paths length constraints to minimize delay
- Ensuring different levels of connectivity for different parts of a heterogeneous VN
  - Can empower a wide variety of Service Level agreements
- Explore possibility of multi-layer augmentation

Thank you

Questions?

Motivation

- A different form of survivability than traditional SVNE
  - Requires no pre-allocated backup path, no path splitting
  - SP reroutes traffic on the failed virtual links to alternate paths
    - Based on traffic priority
    - Thanks to Software Defined Networking (SDN) controller
  - **Connectivity** is required to find alternate paths
- Applicable to VNs carrying **best-effort traffic**
  - May tolerate small amount of delay

CoViNE-ILP Complexity

- Node mapping reduces to finding multiway separator in a graph
  - Poly logarithmic approximation ratio*
- Link mappingextends Multi-Commodity Unsplittable Flow problem
  - Best approximation ratio**:
    - $(7 + \epsilon)$ for line graphs
    - $(8 + \epsilon)$ for cycles
    - Unknown for general graphs
- Disjointedness constraints per conflicting sets increase complexity
  - Best approximation ratio: $L^{\frac{1}{2}-\epsilon}$,L is the number of links***

* Andersen, David G. "Theoretical approaches to node assignment." *Computer Science Department* (2002): 86.
** Bonsma, Paul, et al. "A Constant-Factor Approximation Algorithm for Unsplittable Flow on Paths." *SIAM Journal on Computing* 43.2 (2014): 767-799.
*** Guruswami, Venkatesan, et al. "Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems." *Journal of Computer and System Sciences* 67.3 (2003): 473-496.

CoViNE-Fast Complexity

- Let
  - N = Number of substrate nodes
  - N‘ = Number of virtual nodes
  - L = Number of substrate links
  - L‘ = Number of virtual links
  - σ = Maximum size of location constraint set of any virtual node
  - δ = Maximum degree of a virtual node
- Per link mapping takes $O(L + N \log N)$ time*
- Per node mapping takes $\sigma.\delta.O(L + N \log N)$ time
- Total running time becomes $N'.\sigma.\delta.O(L + N \log N)$

* Fredman, Michael L., and Robert Endre Tarjan. "Fibonacci heaps and their uses in improved network optimization algorithms." *Journal of the ACM (JACM)* 34.3 (1987): 596-615.

CoViNE-Fast algorithm

- Sort virtual nodes on the increasing order of the conflict sets of incident links
- Iterate over virtual nodes in this order
  - Pick the most conflicted node, x
  - Iterate over the candidate node of x
    - Compute minimum cost substrate paths for each virtual link incident to x
  - Map x to the candidate node yielding minimum cost
  - Map a virtual link to its computed path only when both endpoints are mapped