Towards Next Generation Networks with SDN and NFV

Andrea Tomassilli

Coati Team

24-06-2019

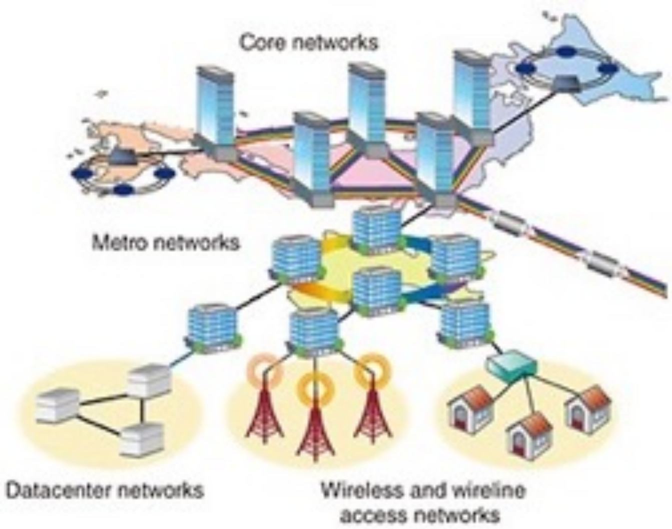
UNIVERSITÉ CÔTE D'AZUR

Outline

- Context of the Thesis
- Research & Contributions
  - Selected Works
  - Additional Contributions
- Conclusion

Context

Optimization of Network Infrastructures



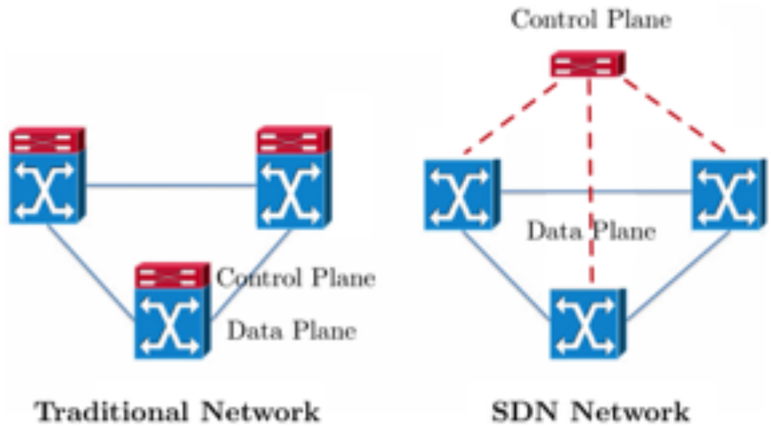
Tools: algorithmic, combinatorics (graph theory), linear programming, simulations, and experimentations

Context
Software Defined Networking (SDN)

Decoupling of network control and forwarding functions

Advantages:

- centralized management
- programmatically configured
- dynamic routing
- ...



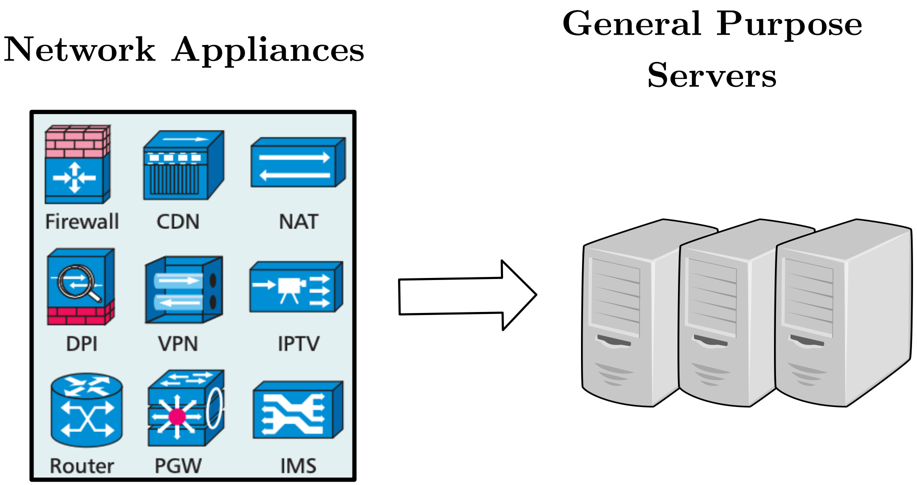
Context

Network Function Virtualization (NFV)

Decoupling of network elements from underlying hardware

Advantages:

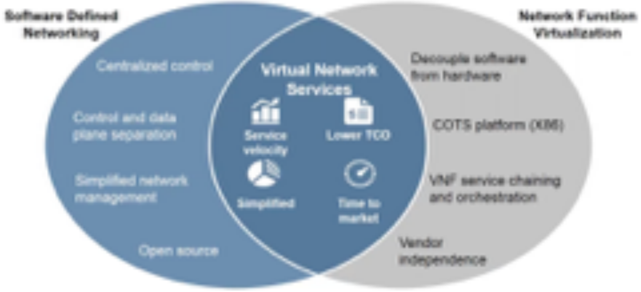
- flexibility
- cost
- scalability
- ...



Context

SDN + NFV => Full Network Programmability

- NFV and SDN **independent of each other** but complementary
- A **symbiosis** between them can improve **resource management** and **service orchestration**:
    - Increased Efficiency and Lower Costs
    - Faster Innovation and Time to market
    - Agility - Automation & change faster
    - No Vendor Lock-in

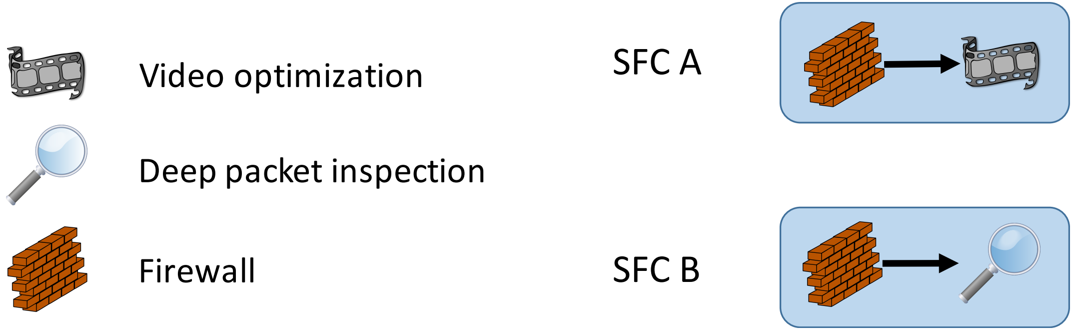


GOAL: exploit the benefits and potentials of both approaches

Use Case

Service Function Chaining (SFC)

- Network flows are often required to be processed by an ordered sequence of network functions
- Different customers can have different requirements in terms of the sequence of network functions



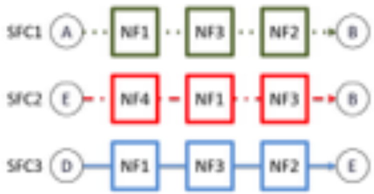
Use Case
Service Function Chaining (SFC)

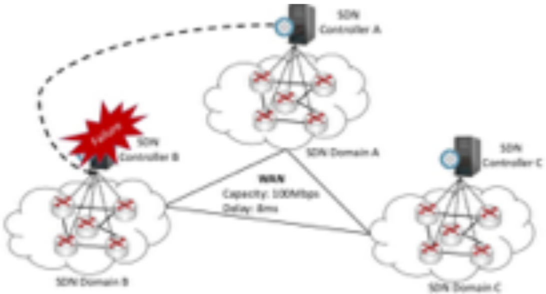
- **Legacy Networks**: new service —> new hardware
  - impractical to change the locations of physical middleboxes
- **SDN/NFV-enabled Networks**: easier and cheaper SFCs deployment and provisioning:
  - simplified middlebox traffic steering (**SDN**)
  - flexible and dynamic deployment of network functions (**NFV**)

Flows can be managed **dynamically from end-to-end** and the network functions can be installed **only along the paths for which and when** they are necessary.

Research Challenges

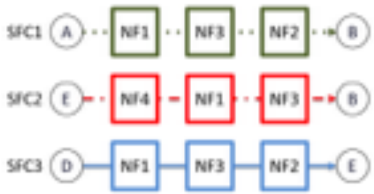
- Algorithmic Aspects of Resource Allocation
- Evaluation of SDN/NFV solutions
- New Protocols & Standardization
- Performance
- Resiliency
- Scalability
- Security
- …

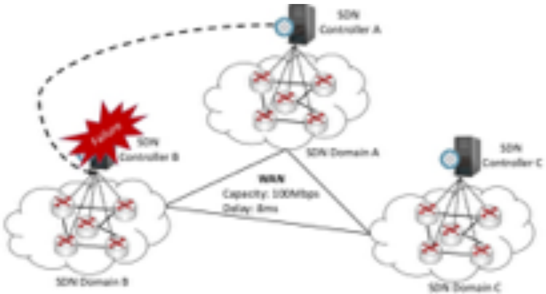




Research Challenges

- **Algorithmic Aspects of Resource Allocation**
- **Evaluation of SDN/NFV solutions**
- New Protocols & Standardization
- Performance
- **Resiliency**
- Scalability
- Security
- …





Outline

- Context of the Thesis
- Research & Contributions
  - Selected Works
  - Additional Contributions
- Conclusion

Contributions

Topic	Chapter	External Collaborations	Publications
Resource Allocation	1	-	*INFOCOM'18*
Survivable SDN/NFV Network Design	2	Diana team, Inria Bordeaux, Orange Labs	*Networking'19 (Poster) + submitted*
Path Protection against link failures	3	Concordia University	*ICC'18*
Energy efficiency	4	-	*INOC'17, JOCN*
Data Center Scheduling	5	-	*INFOCOM'19*
Path Protection in EONs	6	Concordia University	*ONDM'18*
Distributed SDN/NFV Network Experiments	-	Diana team, Orange Labs	*work in progress*

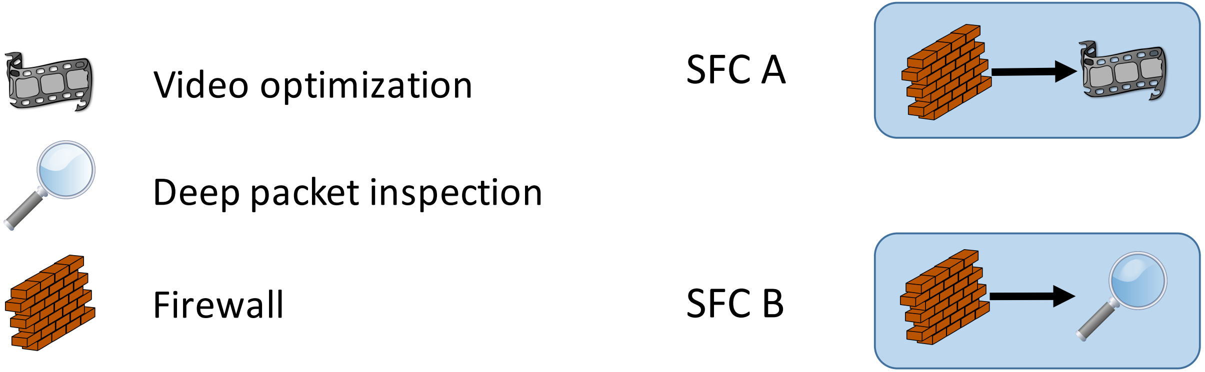
Provably Efficient Algorithms for Placement of Service Function Chains with Ordering Constraints

Andrea Tomassilli, Frédéric Giroire, Nicolas Huin, Stephane Pérennes

IEEE INFOCOM 2018

Preliminaries
Service Function Chaining

Service Function Chain: ordered sequence of network functions to apply to flows on the network

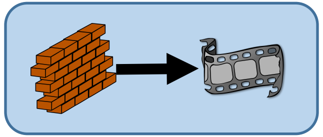


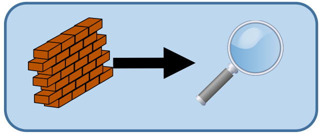
Problem: place VNFs to satisfy the **ordering constraints** of the flows with the goal of **minimizing the total setup cost** (e.g. license fees, network efficiency, or energy consumption)

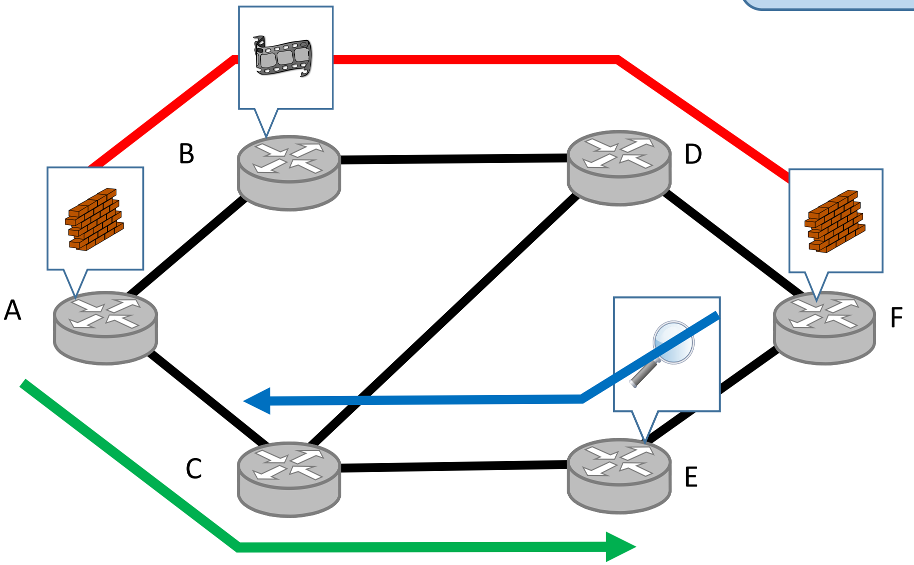
Preliminaries
Example

3 demands: A to F — SFC A

A to E
F to C } SFC B







Related Work

- **Heuristics**
  - [Kuo et al. Infocom 2016] Maximizing the total number of admitted demands -> no warranties
- **ILP based**
  - [Mehraghdam et al. Cloudnet 2014] Minimizing the number of used nodes or the latency of the paths -> problem of scalability
- **Approximation Algorithms**
  - [Cohen et al. Infocom 2015] Minimizing setup cost near-optimal approximation algorithms with theoretically proven performance.
  - [Sang et al. Infocom 2017] Minimizing the total number of network functions. But one single network function.

-> leave the placement of virtual functions with chaining constraint as an open problem.

Contributions

First approximation algorithms taking into account ordering constraints

- **Two algorithms** that achieve a **logarithmic approximation factor**
- Optimal algorithm for **tree network topologies**
- **Numerical results** validate the cost effectiveness of our algorithms

Problem Statement

Input: A digraph $G = (V,E)$, a set of functions F, and a **collection D of demands.**

- A demand $d \in D$ is modeled by a couple :
  - **a path** path(d) of length l(d) and
  - **a service function chain** sfc(d) of length s(d).
- A **setup cost** c(v, f) of function f in node $v \in V$.

Output: A function placement $\Pi \subset V \times F$

Objective: minimize total setup cost

Similarly to [Sang et al. Infocom 2017], we consider the case of an operator which has **already routed its demands** and which now wants to optimize the placement of network functions.

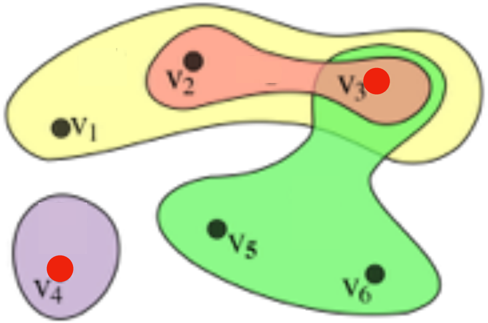
Direct equivalence with the **Minimum Weight Hitting Set Problem**

Input: Collection C of subsets of a finite set S.

Output: A hitting set for C, i.e., a subset S′ ⊆ S such that S′ contains at least one element from each subset in C.

Objective: Minimize the cost of the hitting set

$$\sum_{x \in S'} c_x$$



Preliminaries

Chains of Length 1

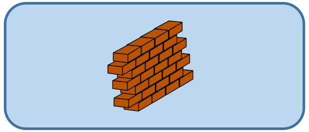
- **Elements of S:** possible **function locations**, i.e., the **vertices** in V. Each element has cost c(v).
- **Sets in C:** **paths** of the demands in D. Set = all path nodes $\{u_1, \ldots, u_{l(d)}\}$.

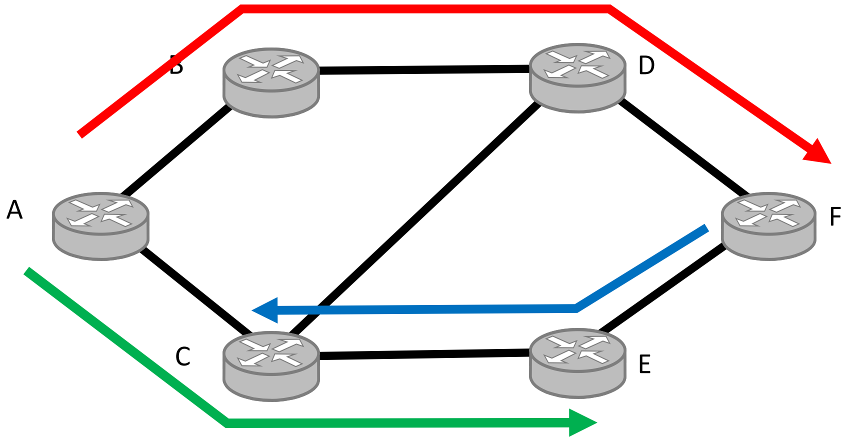
Placement of minimum cost covering all demands corresponds to a minimum cost hitting set.

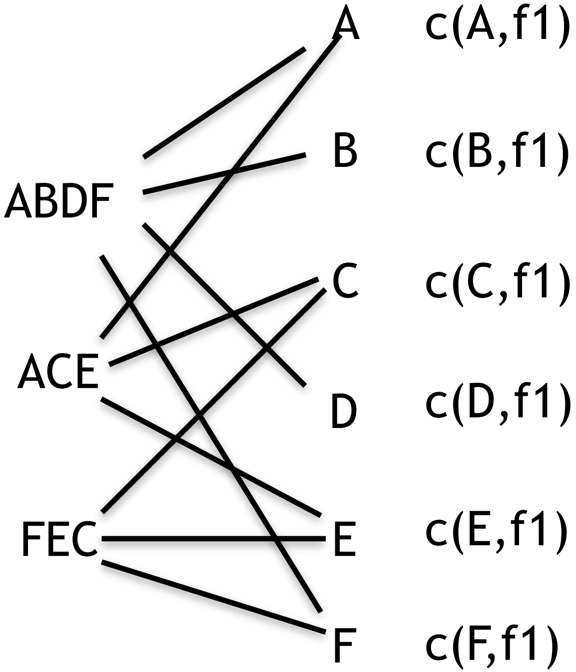
Preliminaries

Chains of Length 1

3 flows: A to F
A to E
F to C
} SFC





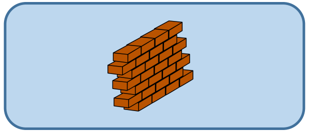


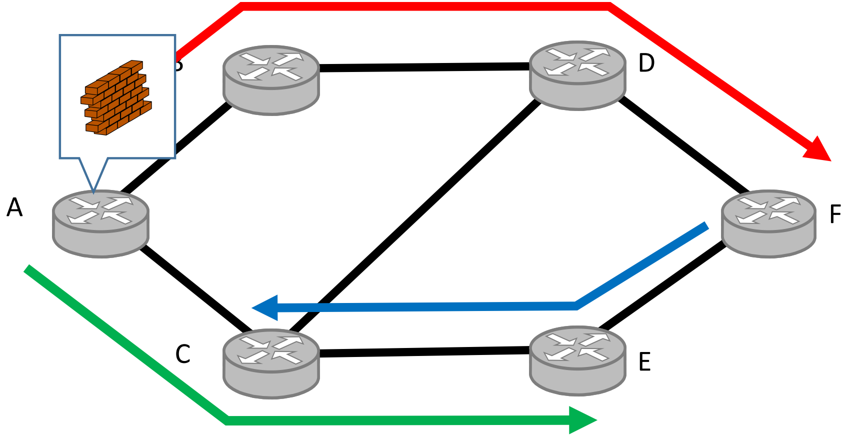
Preliminaries

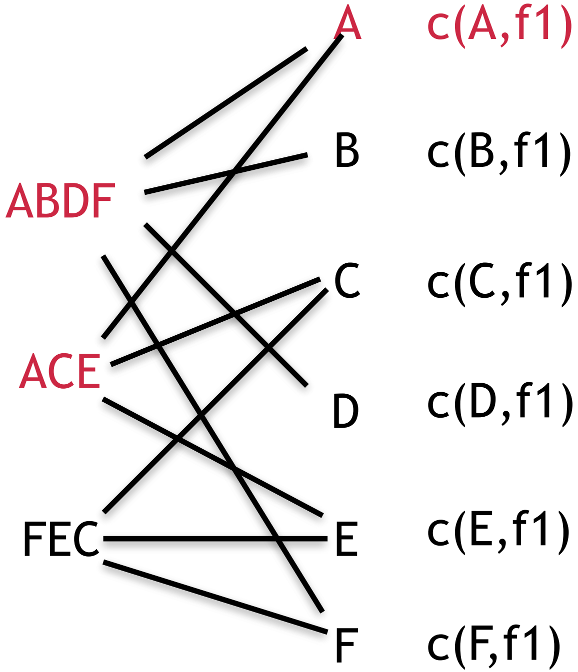
Chains of Length 1

3 flows: A to F
A to E
F to C

} SFC





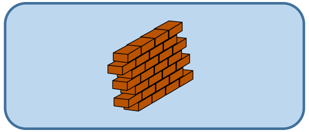


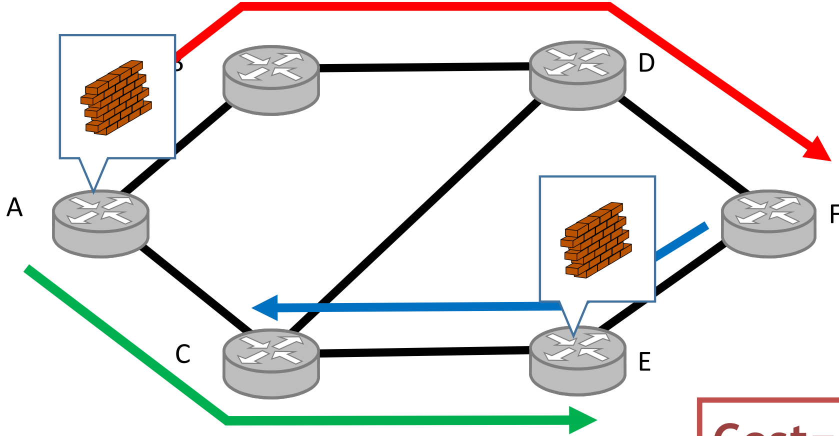
Preliminaries

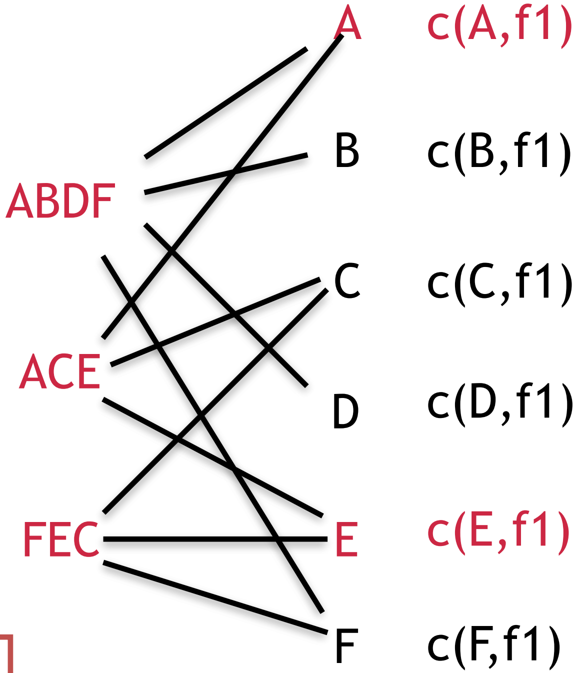
Chains of Length 1

3 flows: A to F
A to E
F to C

} SFC







Cost= c(A,f1)+c(E,f1)

Preliminaries
Chains of Length 1

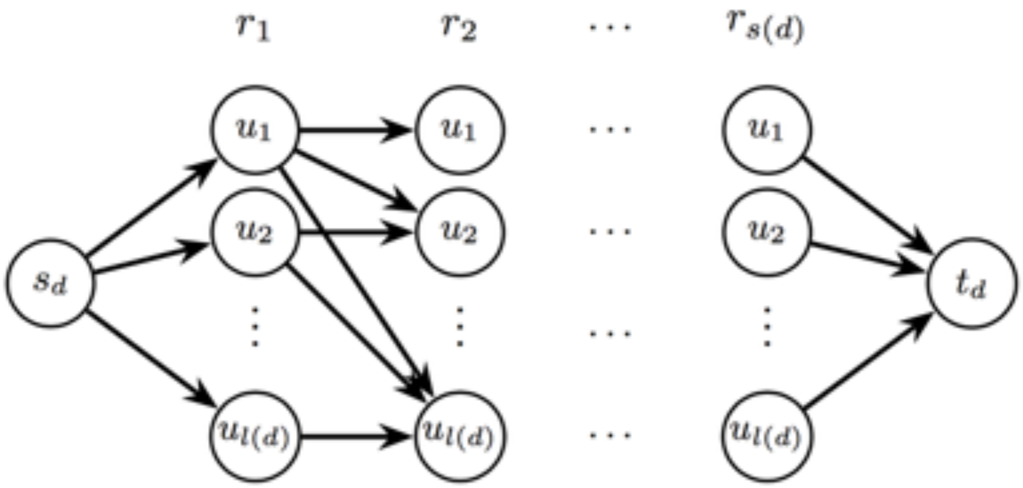
- The equivalence directly gives:
  - On the positive side, an **H(|D|)-approximation** using the greedy-algorithm for Set Cover [Chvatal 1979].
  - On the negative side, SFC Placement Problem is **hard to approximate within ln(|D|)** [Alon et al. 2006].

- When length of the chains ≥ 2,
  Extension is not direct *even for a single chain.*

How to deal with the general case?

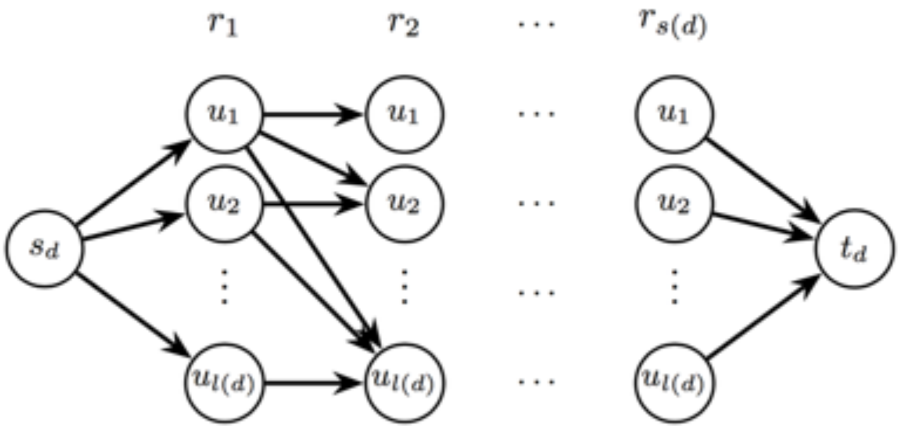
Associated Network

- **Definition:** **Associated Network** H(d) for demand d with path(d) = $u_1, u_2, ..., u_{l(d)}$ and chain sfc(d) = $r_1, r_2, ..., r_s(d)$



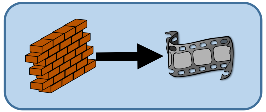
Capacitated Associated Network

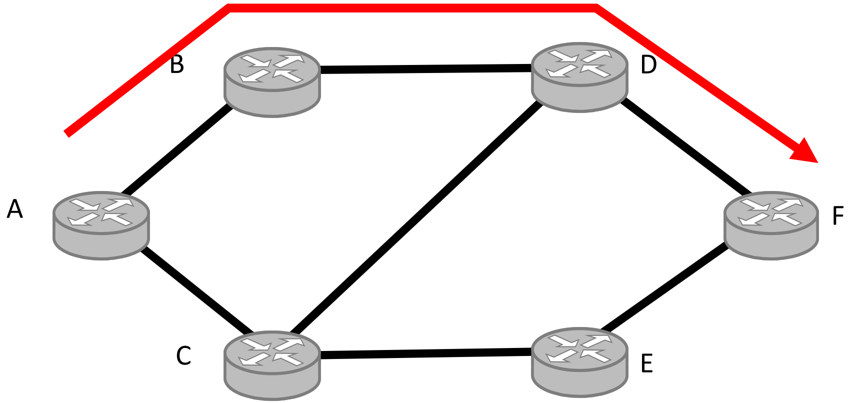
- **Definition: Capacitated** Associated Network H(d,Π) of demand d and **function placement Π**:
  - All arcs have infinite capacity.
  - **Capacity of node u** of layer i is 1 if $(u,r_i) \in \Pi$ and 0 otherwise.

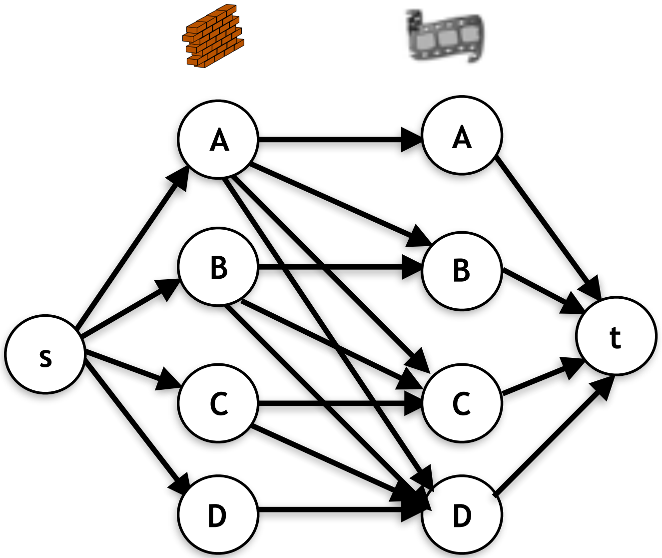


Capacitated Associated Network
An example

1 demand: A to F



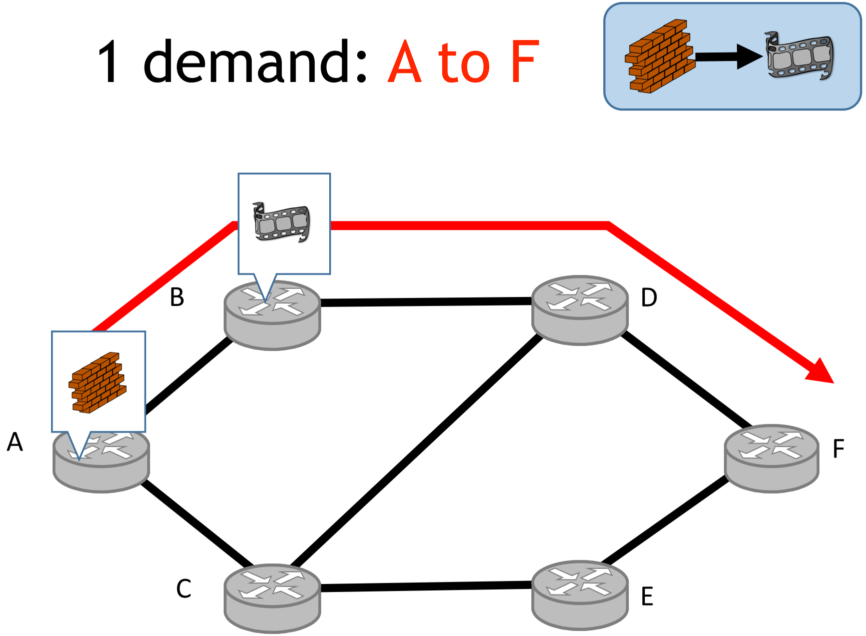


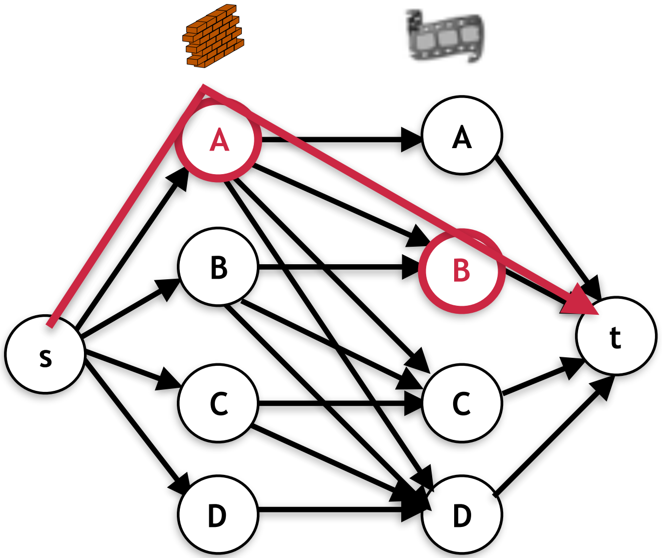


Capacitated Associated Network

An example

1 demand: A to F

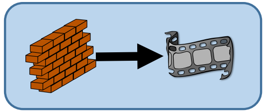


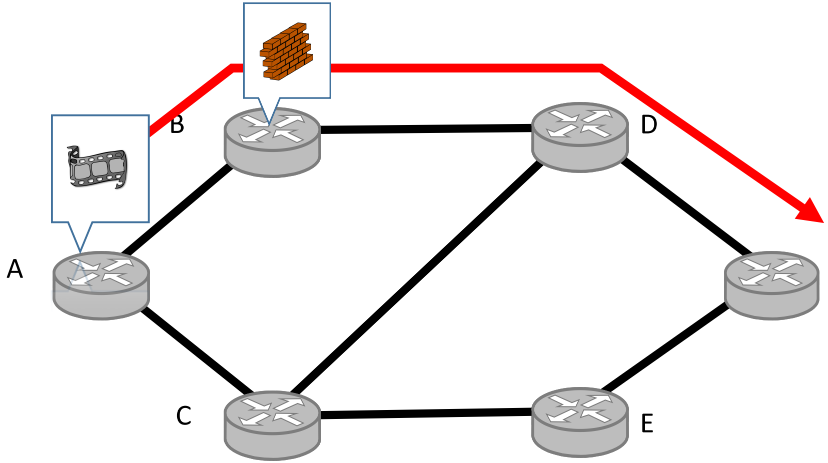


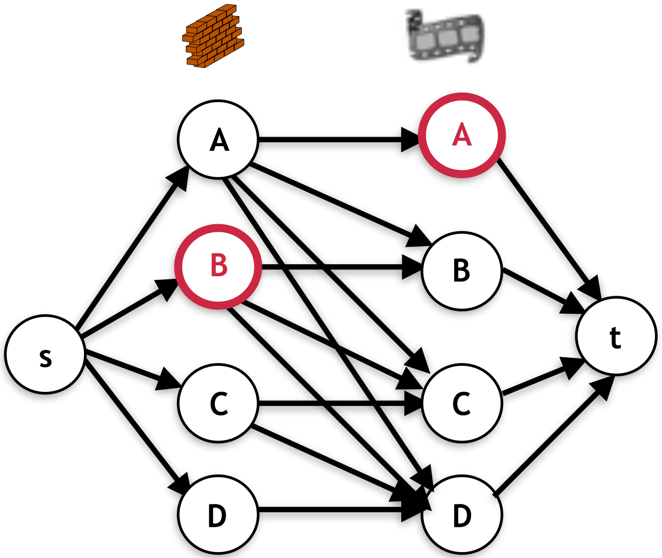
Capacitated Associated Network

An example

1 demand: A to F







Order not respected -> No st-paths

New Formulation of the problem

- **Goal:** *Link our problem with the Hitting Set Problem.*

- **Tool: Menger's theorem** for digraphs (max flow-min cut)
  "*number of st–paths in a digraph is equal to the minimum st-vertex cut*"

Existence of st-paths <=> **cost ≥ 1** of minimum st-vertex cut

New Formulation of the problem

- **Problem: HITTING-CUT-PROBLEM (D,c):**
  - **Elements:** the **function locations (u,f)**, forall u∈ V and f ∈ F. Its cost is c(u,f).
  - **Subsets of the universe: all the st-vertex-cuts** of the associated networks H(d) for all d ∈ D.

-> **find the sub-collection** S of elements (functions placements) hitting all the subsets (cuts) of the universe of **minimum cost.**

Procedure: consider all demands

(1) for each one create the **associated graph**

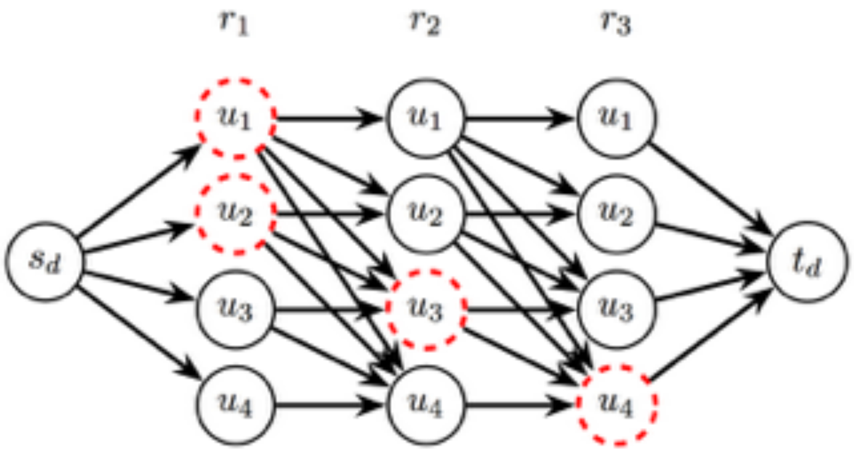
(2) generate all its vertex-cuts

(3) solve the hitting set problem on the **union** **of all the vertex-cuts**

New Formulation of the problem

- **Problem**: **number of cuts exponential in |V|**

-> **Solution:** consider only extremal cuts **(proper cuts)**



Definition 2. A proper st-cut of the associated graph $H(d)$ is a cut of the following form:

$$\{\underbrace{(u_1,1),\dots,(u_{j_1},1)}_{\text{layer }1},\underbrace{(u_{j_1+1},2),\dots,(u_{j_1+j_2},2)}_{\text{layer }2},\dots,$$
$$\underbrace{(u_{j_1+j_2+\dots+j_{s(d)-1}+1},s(d)),\dots,(u_{l(d)=j_1+j_2+\dots+j_{s(d)}},s(d))\}}_{\text{layer }s(d)}$$

for $j_1, j_2, \dots, j_{s(d)} \geq 0$, such that $\sum_{i=1}^{s(d)} j_i = l(d)$.

Number of proper cuts: $\binom{l(d)+s(d)-1}{s(d)-1}$ **Polynomial in |V|**

Approximation Algorithms

Proposition 2. *The problem* SFC-PLACEMENT $(\mathcal{D}, c)$ *is equivalent to a Hitting Set Problem with* $\sum_{d \in \mathcal{D}} \binom{l(d)+s(d)-1}{s(d)-1}$ *sets as an input. If each demand requires at most* $s_{\max}$ *network functions and is associated with a path of length smaller than* $l_{\max}$, *then the size of the instance is at most* $O(|\mathcal{D}| \cdot (l_{\max})^{s_{\max}-1})$.

-> leads to two **approximation algorithms** with **logarithmic factor**

- a **greedy** one (naive and fast versions)
- one using **LP-rounding** (naive and fast versions)

Faster Greedy Algorithm

- **Naive Greedy Algorithm:**

> **While** still not hit proper cuts **do**:
> select function location with ***smallest average cost per newly hit proper cut.***

-> **approximation ratio:** $H(\#\text{Proper Cuts}) = H(|\mathcal{D}|l_{\max}^{s_{\max}-1})$
-> **execution time:** $O(|\mathcal{D}|l_{\max}^{s_{\max}})$

Problem for large chains: if l_{max} in order of network diameter.
Example: Cogent network diameter of 28. For chains of length 10, we would have $\binom{37}{9}$ proper cuts = **124,403,620**

Faster Greedy Algorithm

- Using **specific structure of proper cuts**, provide **much faster greedy algorithm.**

- **Main idea**: **avoid generating all proper cuts.** Only keep track of the number of not hit proper cuts.
- By using **dynamic programming**, number can be counted in time $O(|\mathcal{D}|l_{\max}^2 s_{\max})$ instead of $O(|\mathcal{D}|l_{\max}^{s_{\max}})$

Polynomial in s_{max}

$$N(r,c) = \mathbb{1}_{i^*(r,c)=0} + \sum_{j_c=0}^{r-i^*(r,c)} N(n - j_c, c-1), \text{if } c \geq 2$$

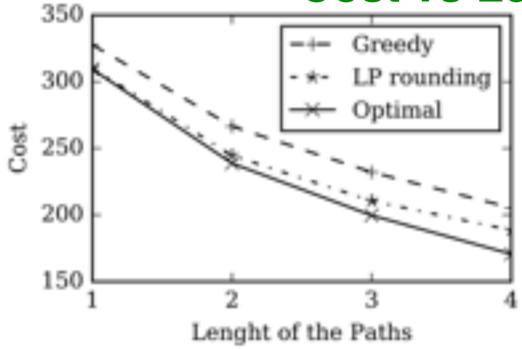
$$N(r,1) = \mathbb{1}_{i^*(r,c)=0}$$

Numerical Results

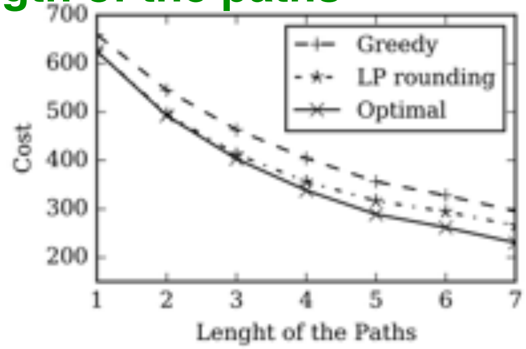
- How **total setup cost** and the **accuracy of our algorithms** vary according to different settings:
  - (i) different path lengths,
  - (ii) increasing number of demands,
  - (iii) increasing length of the chains.
- **Different network topologies:** InternetMCI (19 nodes and 33 links), germany50, (50 nodes and 88 links), and on random Erdös-Rényi graphs.
- Compare the solutions computed by our algorithms with the **optimal ones** computed by IBM ILOG CPLEX.

Numerical Results

Cost vs Length of the paths

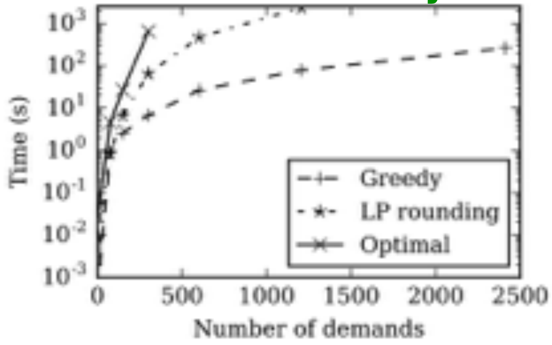


(a) InternetMCI



(b) germany50

Scalability



(1) longer paths -> **more opportunity to share** functions.

(2) logarithmic approximation ratio just a **worst case upper bound.**

(3) **LP rounding** algorithm usually obtains **better ratio** than the **greedy one**, but much **higher processing time.**

Conclusion

- Investigated the problem of **placing VNFs** to satisfy the **ordering constraints** of the flows with the goal of **minimizing the total setup cost.**
- We proposed **two algorithms** that achieve a **logarithmic approximation factor.**
- For the special case of **tree network topologies** we devised an optimal algorithm.
- **Numerical results** validate the cost effectiveness of our algorithms.

Conclusion

- **First theoretical framework** to model the SFC Placement Problem
- **Unaddressed issues:**
  - Accounting of practical constraints such as **soft capacities** on network functions or **hard capacities** on network nodes.
  - Affinity/anti-affinity rules
  - Partial order
  - Latency

Future research direction: possible to efficiently approximate these problems?

Bandwidth-optimal Failure Recovery Scheme for Robust Programmable Networks

A. Tomassilli, G. Di Lena, F. Giroire, I. Tahiri, D. Saucez, S. Perennes, T. Turletti, R. Sadykov, F. Vanderbeck , C. Lac

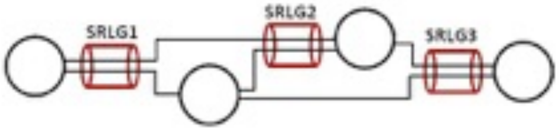
IEEE/IFIP Networking 2019 (Poster) + submitted

Motivation

- Nodes and Links are **failure-prone** [1]
- Network failures such as (multiple) link or node failures may have a significant impact on the QoS experienced by the customers and to SLA violations

Resiliency needs to be addressed while designing a network

- Failures tend to be correlated between them (-> SRLGs) [2]



[1] D. Turner, et al. "**California fault lines: understanding the causes and impact of network failures**." ACM SIGCOMM 2010

[2] S. Kandula et al. “**Shrink: A tool for failure diagnosis in ip networks**,” ACM SIGCOMM workshop 2005

Introduction
Fault management techniques

- Different **approaches** to deal with network failures:
  - **protection** vs **restoration**
  - **link** vs **path** protection
  - **dedicated** vs **shared** protection
  - …

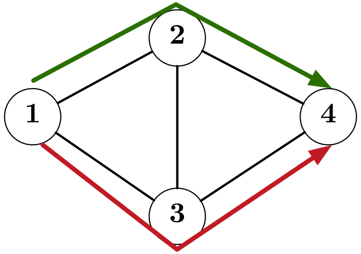
Which solution is the **best one**? Several tradeoffs

- recovery guaranteed vs efficient bandwidth utilization
- local vs end-to-end repair
- low implementation complexity vs low bandwidth overhead

Introduction
Global Rerouting

Example: 1 demand from Node 1 to Node 4

Classic Path Protection

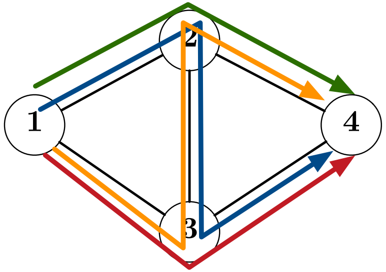


No failure
Failure (1,3)
Failure (3,4)
Failure (2,3)
Failure (1,2)
Failure (2,4)

- 2 paths: primary and backup
- if a failure affects the primary path

  -> move to the backup path

Global Rerouting



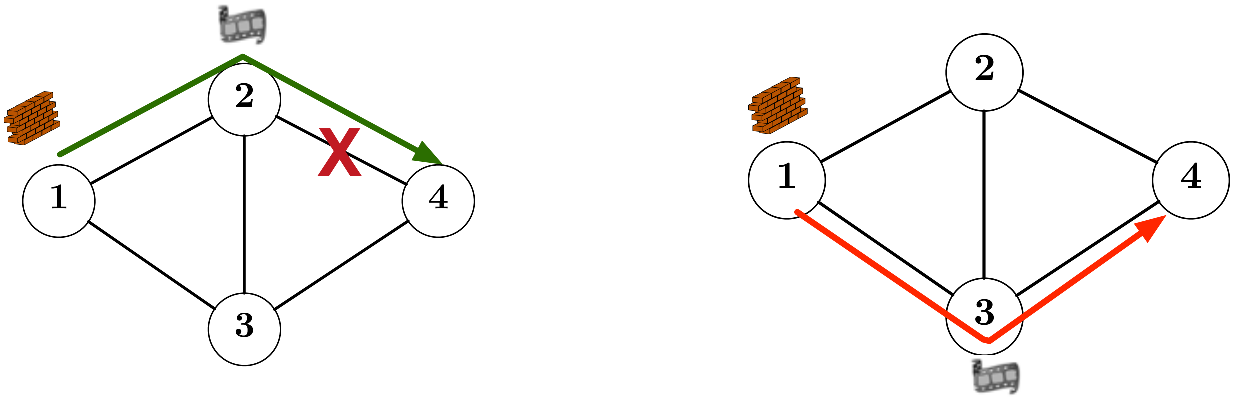
No failure
Failure (1,3)
Failure (3,4)
Failure (2,3)
Failure (1,2)
Failure (2,4)

- There could potentially be a path for each failure situation

Introduction

Global Rerouting **with SFC constraints**

- SFCs add **additional constraints**:
  1. The **execution order** must be guaranteed in all the paths for a demand



  2. A node may be able to run **only a subset** of VNFs (which might be empty)

Introduction

Global Rerouting with SFC constraints

- For each failure situation a **completely different routing** for the demands
  - bandwidth **optimality (due to shared backup resources)**
  - large number of rules to install on the network devices
    -> **huge signaling overhead**

Impractical on legacy networks

How the **introduction of SDN and NFV** change the game?

Problem Statement

Given:

- an undirected graph G = (V, E)
- a set D of requests with **source**, **destination, SFC, and bandwidth demand**

The problem consists in finding:

- a **primary** paths provisioning
- a **backup** paths provisioning for each SRLG failure

} SFCs provisioning must be guaranteed

GOAL

Minimize the total bandwidth requirements

Related Work

Using a set of pre-configured multiple backup **not new.**

1) **Multiple pre-configured paths** [Suchara et al. SIGMETRICS'11]
2) **Small set of backup routing configurations** to be kept in the routers [Kvalbein et al. INFOCOM '06, INFOCOM '07]
3) **OpenFlows Fast Failover Group Tables** to quickly select a preconfigured backup path in case of link-failure [Sgambellini et al. JOCN '13]

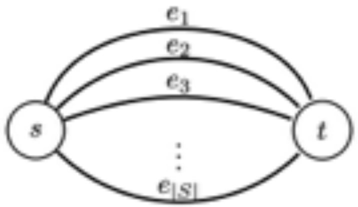
Idea of using multiple routing configurations **to the extreme**
a potentially completely different routing after a failure

Problem Complexity

- **Dimensioning** problem for an ISP who wants to minimize resource usage —> **No capacities**
- The problem may appear easy but …

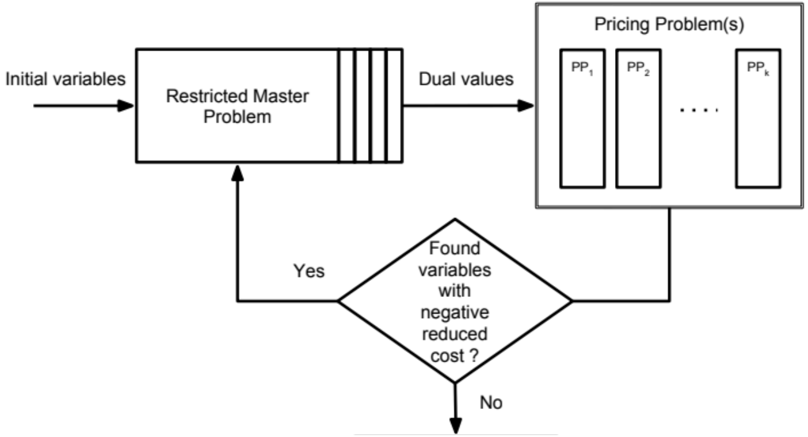
Proposition 1. *The* GLOBAL REROUTING *problem is NP-hard even for a single demand, and cannot be approximated within*
$(1-\epsilon)\ln(|R|)$ *for any* $\epsilon > 0$ *unless P=NP, where* $|R|$ *denotes the number of failing scenarios.*

Tool: Reduction from the Hitting Set Problem

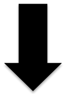


Optimization Model: Column Generation

Decompose the original problem into a **restricted master problem** (RMP) and several **subproblems** (Pricing Problems)



(fractional) optimal solution found



Integer solution

Column Generation

Master Problem

x_{uv}: bandwidth to be deployed on link (u,v)

$$\min \sum_{(u,v)\in E} x_{uv}$$

OBJECTIVE: minimization of the required bandwidth

(1) A path for each demand d and SRLG failure situation r

$$\sum_{\pi\in\Pi_d^r} y_\pi^{d,r} \geq 1 \qquad (1)$$

(2) Bandwidth utilization for each link (max utilization considering every failure scenario)

$$x_{uv} \geq \sum_{d\in\mathcal{D}} \sum_{\pi\in\Pi_d^r} bw_d \cdot a_{uv}^{\pi} \cdot y_\pi^{d,r} \qquad (2)$$

Column Generation
Pricing Problems

- A subproblem for **each demand and SRLG failure situation**
- Objective function:

$$\min -\alpha_r^d + bw_d \cdot \sum_{(u,v)\in E} \beta_{uv}^r \cdot \sum_{k=0}^{\ell(d)} \varphi_{(uk,vk)}$$

dual variables of constraints (1)

dual variables of constraints (2)

The pricing subproblem can be reduced to a **weighted shortest-path problem** with link weight β_{uv}^r

⬇

It can be solved using a SP algorithm (e.g., Dijkstra)

Column Generation Last Step

From Splittable (Fractional) to Unsplittable (Integer)

- A solution corresponds to an optimal set of **fractional** flows

GOAL: one **single unsplittable path** for each demand and SRLG failure situation

Classical Approach: change the domain of the variables in the last RMP from continuous to integer and use an ILP solver

Column Generation Last Step

From Splittable to Unsplittable

Our strategy: choose a **path per demand and failure situation** while minimizing the **overflow** (additional capacity) to be allocated in the network

The Min Overflow Problem:

Input: A digraph G = (V, E, **c***), a collection D of demands**,** each associated with **a set of paths** (from the fractional solution)

Output: A path per demand and failure situation, new capacities $\tilde{\mathbf{c}}$

Objective: minimize the overflow $\sum_{(u,v)\in E} \frac{\tilde{c}(u,v)}{c^*_{uv}}$

The min overflow problem

Approximation Algorithm

Proposition 2. *The* MIN OVERFLOW PROBLEM *is APX-hard (and so is NP-Hard) and cannot be approximated within a factor of* $1 + \frac{3}{320}$, *unless P=NP.*

Tool: Reduction from MAX 3-SAT

Proposition 3. *The* MIN OVERFLOW PROBLEM *can be approximated with high probability within a factor of* $(1 + \frac{1}{e}) + \epsilon$, *for any* $\epsilon > 0$.

Tool: Randomized Rounding

Numerical Results

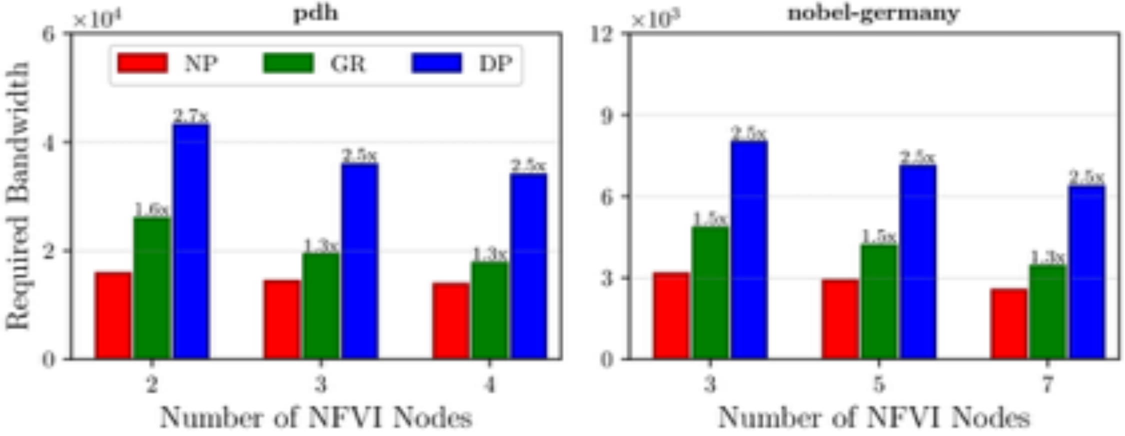
- Simulations on both **real** (SNDLib) and **random generated** instances

Classical Approach		Approximation Algorithm	
time	ϵ	time	ϵ
11mn	4%	40s	12.7%
40s	0.22%	20s	1.4%
40s	0.17%	30s	3.2%
50s	0.3%	10s	5.5%
1mn	0.6%	30s	2.7%
6mn	0.2%	1mn	4.5%
27mn	2.2%	2mn	9.2%

Tradeoff: accuracy vs computational time

Numerical Results

- Comparison of Global Rerouting (**GR**) with Dedicated Path Protection (**DP**) and no protection (**NP**)



Global Rerouting requires **only** between 30% and 60% more bandwidth

Dedicated Path Protection requires **almost 3 times** more bandwidth

Experimental Evaluation

A new route for all the demands is possible after a failure takes place

⬇

Large number of rules to be changed
on the network devices

Can the Global Rerouting be put in practice?

Mininet as a tool to answer the question

Experimental Evaluation

Experimental Setup:

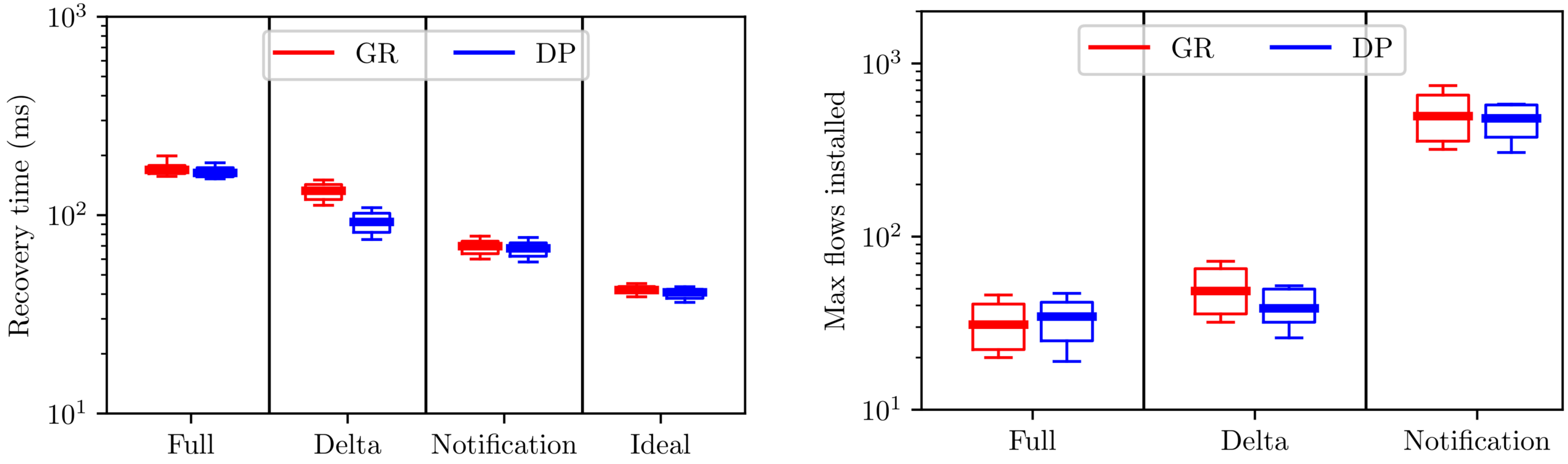
- **Mininet** to emulate the network
- **OpenDayLight** as a SDN Controller
- Routing tables computed by the optimization algorithm
- Simulated a failure and looked at the time to reroute

3 possible implementations:

- **Full**: re-installation of all rules in the switches by the controller
- **Delta**: installation by the controller of rules for the flows which have changed
- **Notify**: pre-installation of all the rules in the switches and notification sent to the switch whose links are down

Experimental Evaluation

polska: 12 nodes, 18 links, and 66 demands



Tradeoff: recovery time vs flow table sizes

Small overhead wrt a classical Dedicated Path Protection scheme

Bandwidth optimal failure recovery **may be achieved** thanks to SDN

Mininet
Limitations

- **Mininet** is a network emulator supporting OpenFlow based Software-defined Networking (SDN)
- Provides a flexible and cost-efficient experimental platform to evaluate SDN applications but it has several **limitations**:
  - **resources limits** (CPU, bandwidth) if experiments are run on a single host:
  - no strong notion of **virtual time** (timing measurements based on system clock)

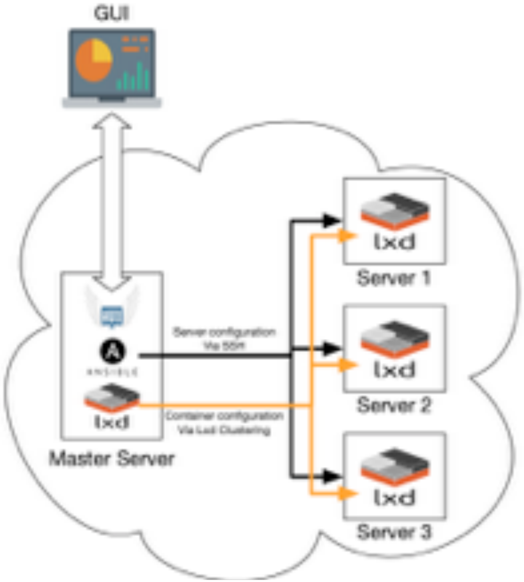
⬇

A new software tool **Distrinet**

A solution to **overcome** Mininet Limitations and **increase** the performance fidelity of network experiments

Distrinet
Key Features

1. Able to **distribute experiments** on either Cluster or public clouds (Amazon EC2)
2. **Compatible** with Mininet (same APIs)
3. **Optimize** Resource Allocation solving:
   - a Virtual Network Embedding Problem (Private Clusters)
   - a Vector Bin Packing with Multiple-Choice (Public Clouds)



Work in progress with Diana (Inria Team) & Orange Labs.

Conclusion

- Studied the ISP network dimensioning problem with protection against a Shared Risk Link Group failure
- Optimization model based on a column generation approach
- LP and approximation-guaranteed greedy based heuristics validated numerically
- Evaluated several implementation options
- A new software tool to overcome Mininet limitations

Outline

- Context of the Thesis
- Research & Contributions
  - Selected Works
  - Additional Contributions
- Conclusion

Additional Contributions

Other work on SFC Provisioning Problem

france telecom
3.7 TWh ~ 0.5 billion US$

verizon
Corporate Responsibility
9.9 TWh ~ 1.5 billion US$

- **Problem #1**:

  Telecom infrastructure and devices account for 25% of ICT's energy consumption.

  How to improve energy efficiency using SDN+NFV?

  N. Huin, A. Tomassilli, F. Giroire, B, Jaumard. **Energy-Efficient Service Function Chain Provisioning**, IEEE/OSA Journal of Optical Communications and Networking.

- **Problem #2**:

  Classical Path Protection techniques do not deal with SFC contraints and NFVI nodes

  How to extend classical path protections schemes to deal with SFC constraints?

  A. Tomassilli, N. Huin, F. Giroire, B. Jaumard**, Resource Requirements for Reliable Service Function Chaining,** IEEE ICC'2018

- **Tools**:

  Optimization (Column Generation), Heuristics, Simulations

Additional Contributions

How to schedule jobs in a data center taking into account network transfer?

- **Context**:

  Today's most common applications spend a significant portion of their time in communications (up to 50% of the competition time)

- **Problem**:

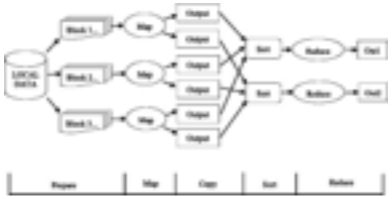
  How to to model the orchestration of tasks in a datacenter for scenarios in which the network bandwidth is a limiting resource?

- **Tools**:

  Optimization (Approximation Algorithms, Heuristics), Simulations on Google Trace

> Cluster computing applications like MapReduce and Dryad transfer massive amounts of data between their computation stages. These transfers can have a significant impact on job performance, accounting for more than 50% of job completion times. Despite this

[SIGCOMM Orchestra 2011]



F.Giroire, N. Huin, A. Tomassilli, S. Pérennes. **When network matters: Data center scheduling with network tas**
IEEE INFOCOM 2019.

Additional Contributions

How to protect demands from failure in an Elastic Optical Network?

- **Context**:

  EONs are based on a flexible spectrum allocation with the aim of overcoming the fixed, coarse granularity of existing WDM technology

- **Problem**:

  Provide path protection against SRLG failures to the lightpaths minimizing the spectrum requirements

- **Tools**:

  Optimization (Column Generation), Simulations

A. Tomassilli, B. Jaumard, F.Giroire. **Path Protection in Optical Flexible Networks with Distance-adaptive Modulation Formats**, ONDM 2018.

Outline

- Context of the Thesis
- Research & Contributions
  - Selected Works
  - Additional Contributions
- Conclusion

To conclude …

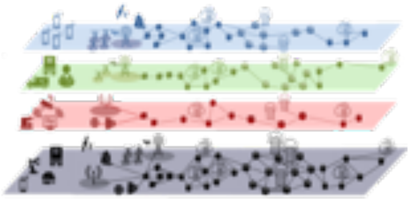
- **SDN** and **NFV** bring several **benefits**:
  - simplify management
  - enhance flexibility of the network
  - reduce the network cost
- But also several **challenges** that need to be addressed to fully attain their benefits

SDN-NFV enabled network has the potential to boost NFV deployment and support new efficient and cost-effective services

Possible Future Directions

- Network Reconfiguration
- Assign slices to capacity slots of physical links -> **slicing**
- Dynamic SFC Placement
- Several major revolutions:
  - 5G
  - IoT
  - Mobile Edge Computing
  - …

} New challenges



New algorithmic problems to be solved

Thank you for your attention

Backup

Fog/Edge Computing

- **PROBLEM**: Interactive applications require ultra-low network latencies (< 20 ms) … but latencies to the closest data center are 20-30 ms using wired networks, up to 50-150 ms on 4G mobile networks
- **SOLUTION**: **Exploit distributed and near-edge computation**:
  - Reduce latency and network traffic
  - improve power consumption
  - increase scalability and availability

FOG COMPUTING

Analyze most IoT data near the devices that produce and act on that data

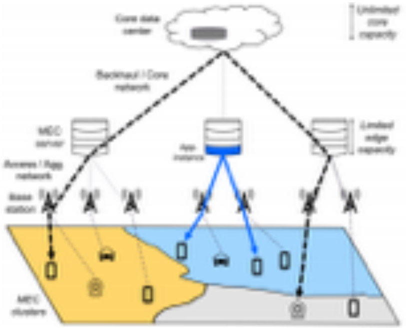
Fog/Edge Computing - Challenges

How to assign the IoT applications to computing nodes (fog nodes) which are distributed in a Fog environment?

- Computing and networking resources are:
  - heterogeneous
  - not always available
- Service cannot be processed everywhere
- Demands and resources are dynamic

Mobile Edge Computing

- **IDEA**: Offloading to improve latency and alleviate congestion in the core -> Push the content (application servers) close to the users using **MEC servers** (small datacenter collocated with the base station) in the infrastructure close to the edge of the network
- **PROBLEM:** assign users, application, and share of traffic to the MEC servers
- **Constraints:**
  - mobile traffic depends on time and locality
  - geographically constraints
  - mobility of the users
  - budget



Dealing with Node Failures

- Failure of a VNF (attacks, software errors) can induce the failure of all the service
- Different approaches
  - **k-resiliency**: k slaves for each VNF
  - **availability**: guarantee high levels of VNF availability

  -> probability that a the network function is working at

  a given time

Network Slicing

s, t, bw, latency

- Assign slices to capacity slots of physical links
  - each slice is independent from each other
  - each slice may have different QoS requirements
- 2 different network slicing strategies:
  - **SOFT**: traffic is multiplexed in queuing systems: high load may affect other slices
  - **HARD**: each slice has dedicated resources at physical and MAC layers