Breaking Service Function Chains with *Khaleesi*

Sara Ayoubi,
Shihabur R. Chowdhury,
Raouf Boutaba

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

Service Function Chaining (SFC)

Network policies often require packets to traverse an ordered set of Network Functions

Service chain for a Web Service

Service chain in enterprise Data Center Networks

Service Function Chaining (SFC)

Network policies often require packets to traverse an ordered set of Network Functions

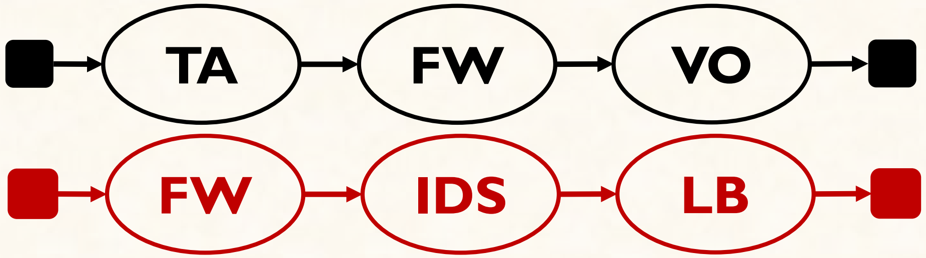
Service chain for a Web Service

Service chain in enterprise Data Center Networks

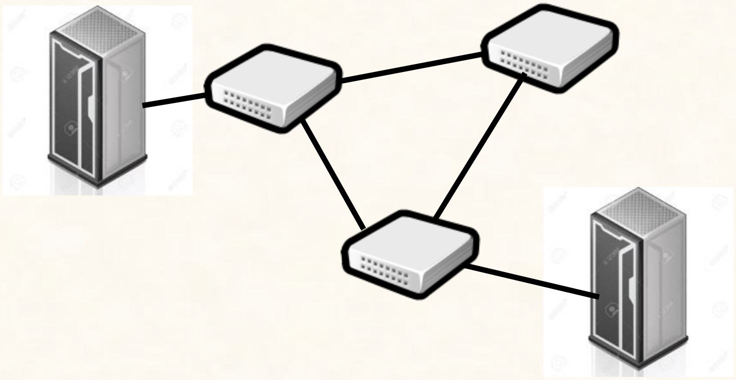
Traditionally, network functions have been realized by hardware middleboxes

SFC Orchestration in NFV

Given



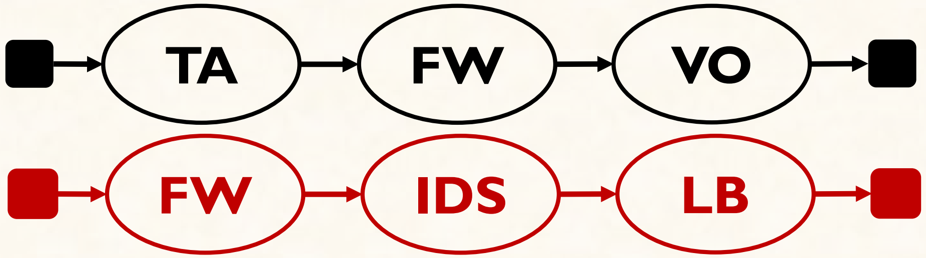
A set of SFC requests



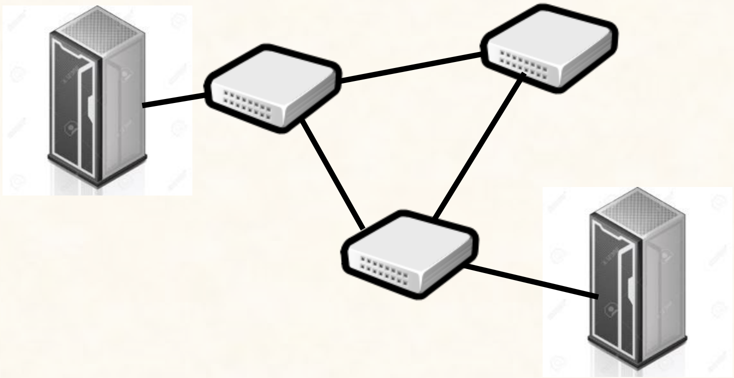
A set of Physical Resources
(Servers, Switches, Links)

SFC Orchestration in NFV

Given



A set of SFC requests



A set of Physical Resources
(Servers, Switches, Links)

Determine

Placement of NFs on Servers

Traffic Routing between VNFs

Objective: Minimize OPEX/ Energy Cost/Network B.W, or Maximize Acceptance Ratio *etc.*

SFC Orchestration: State-of-the-art

O(100) research papers on the topic

Different aspects have been considered

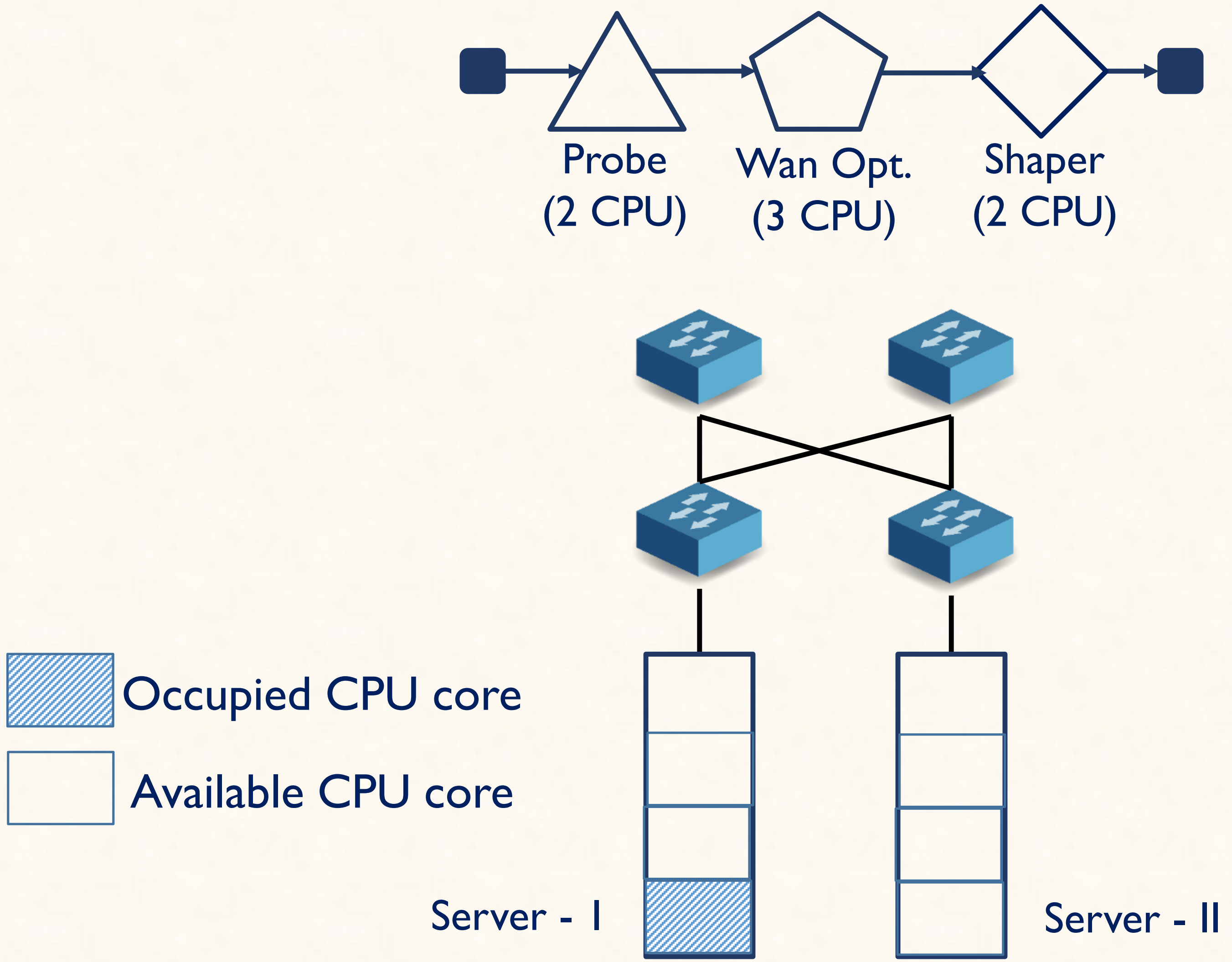
Minimization of energy, network bandwidth, SLO violation, *etc.*
Maximization of availability, acceptance ratio, *etc.*

In most cases, the SFC is considered *rigid*, *i.e.*, *traversal order of the NFs cannot be modified at all*

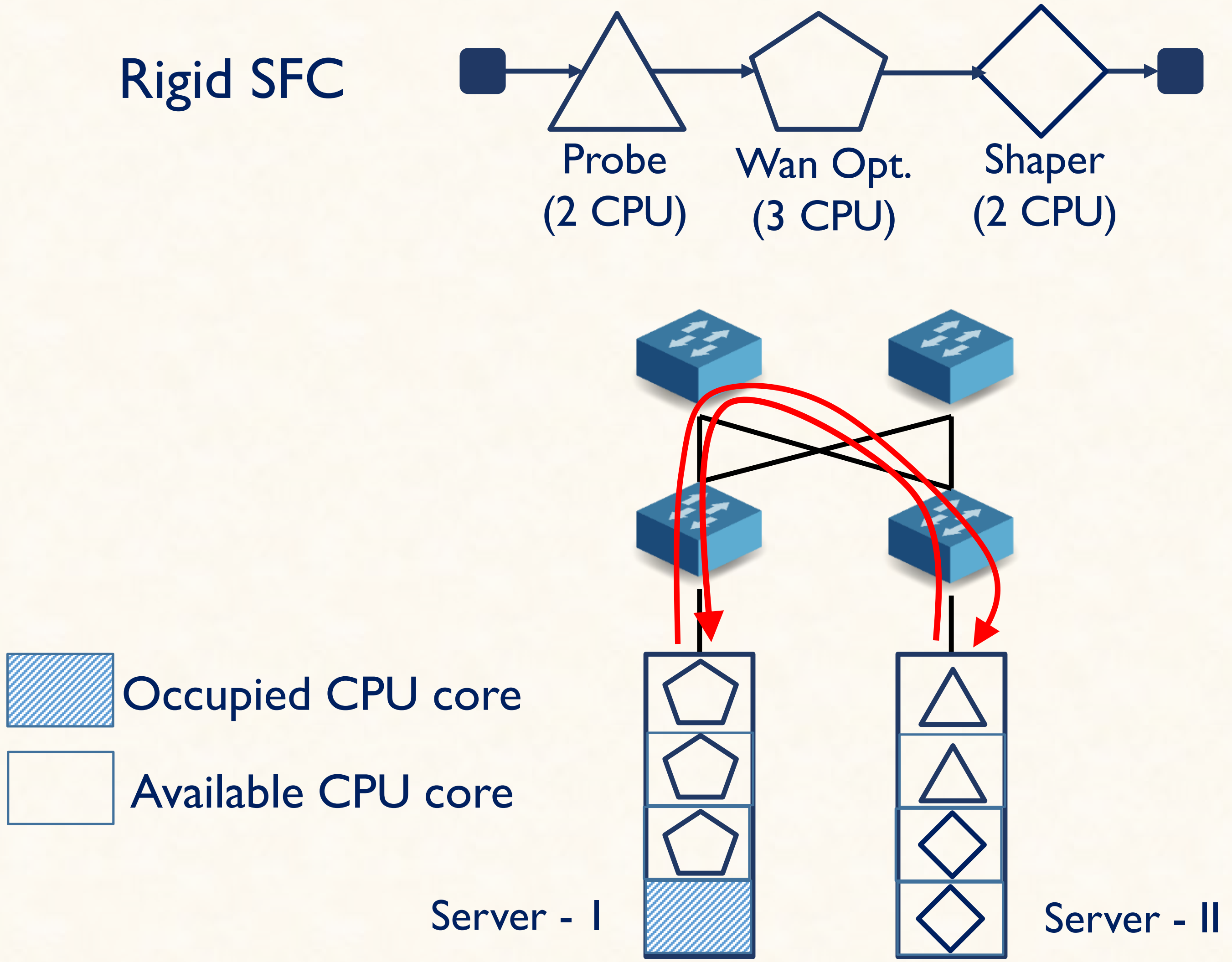
Question:

Can we perform better resource allocation *if we break the traversal order* in an SFC?

Illustrative Example



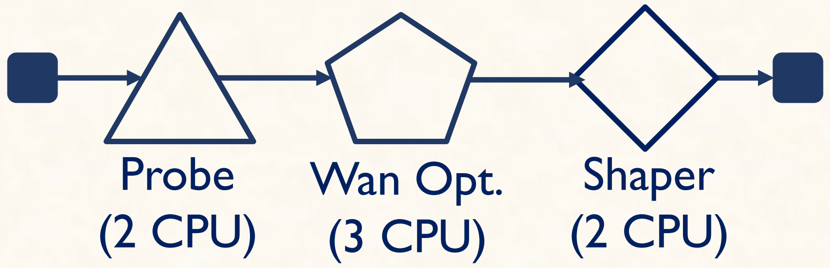
Resource Allocation for Rigid SFC

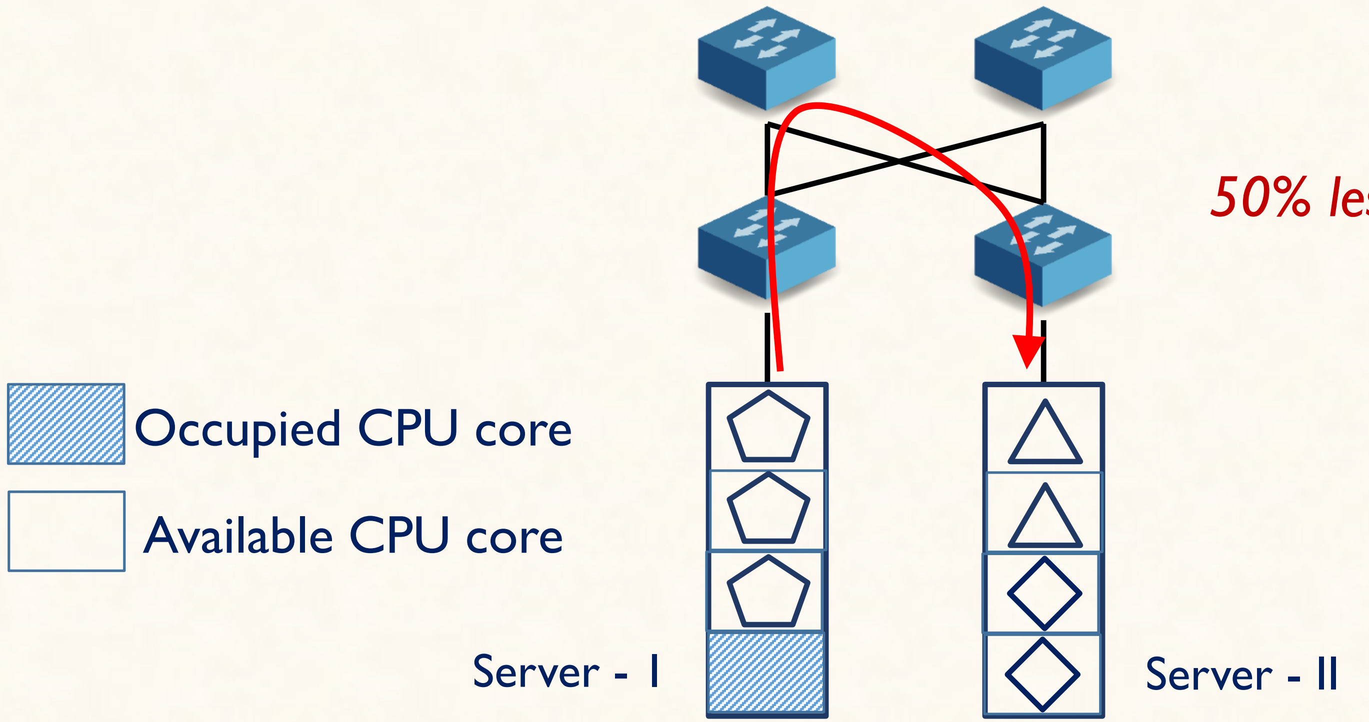


Resource Allocation for Flexible SFC

Flexible SFC

(Probe and Wan Opt. can be swapped)





50% less network hops

More Questions

How can we tell if NF traversal order can be *broken*?

How to *exploit* flexible SFCs in resource allocation?

How much *benefit* can we get from flexible SFCs?

Our Contributions

Question	Contribution
How can we tell if NF traversal order can be *broken*?	Theoretical analysis for detecting *re-order compatible* NFs
How to *exploit* flexible SFCs in resource allocation?	*Khaleesi**: Suit of solutions for SFC Orchestration, *while breaking NF traversal order*
How much *benefit* can we get from flexible SFCs?	Empirical evaluation of *benefits*

*Character from the popular fantasy novel *A Song of Ice and Fire* (adapted as TV series *Game of Thrones*), who is also known as *the breaker-of-chains*

Flexible SFC: State-of-the-art

Language and Data Model[1,2]

Flexibility is part of input; does not demonstrate any quantifiable benefit.

NF Execution Parallelization[3,4]

Requires additional specialized components

Flexibility is determined from NF properties for finding Optimal Solution; empirically evaluated quantifiable benefits demonstrated

1. S. Mehraghdam *et al.*, "Placement of services with flexible structures specified by a yang data model," IEEE NetSoft, 2016
2. S. Mehraghdam *et al.*, "Specifying and placing chains of virtual network functions," IEEE CloudNet, 2014
3. Y. Zhang *et al.*, "Parabox: Exploiting parallelism for virtual network functions in service chaining," ACM SOSR, 2017
4. C. Sun et al., "Nfp: Enabling network function parallelism in nfv," ACM SIGCOMM, 2017

Flexible SFC: State-of-the-art

Language and Data Model[1,2]

Flexibility is part of input; does not demonstrate any quantifiable benefit.

NF Execution Parallelization[3,4]

Requires additional specialized components

In our case

Flexibility is determined from NF properties for finding Optimal Solution; empirically evaluated quantifiable benefits demonstrated

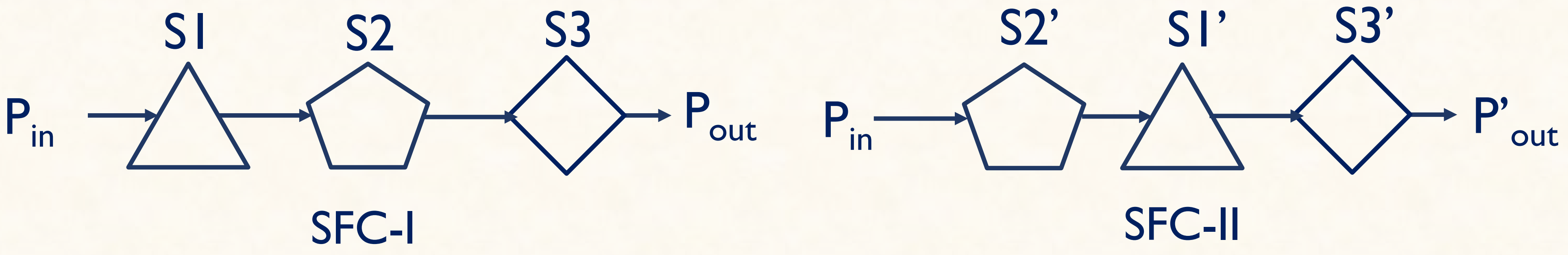
1. S. Mehraghdam *et al.*, "Placement of services with flexible structures specified by a yang data model," IEEE NetSoft, 2016
2. S. Mehraghdam *et al.*, "Specifying and placing chains of virtual network functions," IEEE CloudNet, 2014
3. Y. Zhang *et al.*, "Parabox: Exploiting parallelism for virtual network functions in service chaining," ACM SOSR, 2017
4. C. Sun et al., "Nfp: Enabling network function parallelism in nfv," ACM SIGCOMM, 2017

How to determine *re-order compatibility*?

When is a pair of NFs re-order compatible?

When swapping their order in an SFC results in *semantically equivalent* SFCs

What is *semantic equivalence*?



SFC-I & SFC-II are *semantically equivalent iff*
$P'_{out} = P_{out}$ and (S1,S2,S3) =(S1',S2',S3')

Conditions for Re-order Compatibility

NF Operation involves:	Reads/Write *Packet Headers* → *Interest fields*
	Update internal state based on *packet header* → *State fields*

Conditions for Re-order Compatibility

NF Operation involves:

Reads/Write *Packet Headers* → *Interest fields*

Update internal state based on *packet header* → *State fields*

Two NFs are reorder compatible *iff*

Their state fields are disjoint and interest fields are either disjoint or read-only

Conditions for Re-order Compatibility

NF Operation involves:

- Reads/Write *Packet Headers* → *Interest fields*
- Update internal state based on *packet header* → *State fields*

Two NFs are reorder compatible *iff*

Their state fields are disjoint and interest fields are either disjoint or read-only

	Firewall	Proxy	IPS	Shaper	NAT	DPI	WANX	Probe
Firewall			✓	✓		✓	✓	
Proxy			✓	✓		✓		
IPS	✓	✓		✓				✓
Shaper	✓	✓	✓			✓	✓	
NAT							✓	
DPI	✓	✓		✓				✓
WANX	✓			✓	✓			✓
Probe			✓			✓	✓	

Re-order compatibility matrix of commonly used enterprise NFs

Reorder Compatibility + SFC Orchestration

Re-order Compatibility + SFC Orchestration
= *Flexible SFC Orchestration*

Khaleesi: A suit of solutions to Flexible SFC Orchestration

OPT-Khaleesi

ILP-based Optimal Solution
(NP-hard)

FAST-Khaleesi

Greedy Heuristic

Assumptions

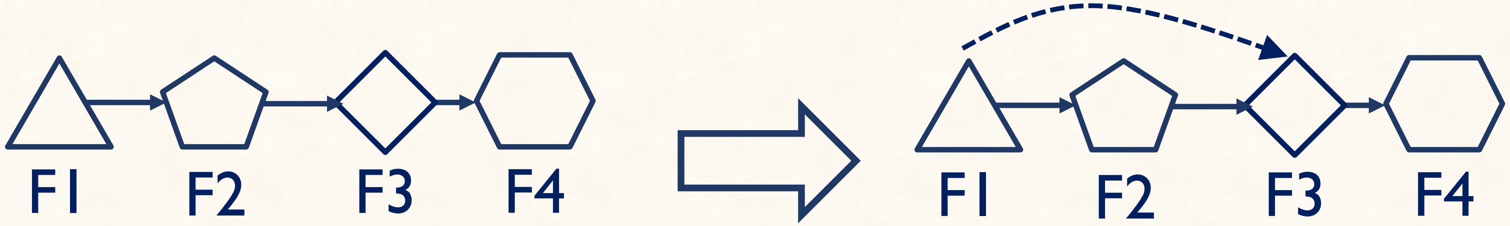
Linear Chain of NFs (no branches)

No change of data rate after swapping

Heterogeneous servers

OPT-Khaleesi*

Augment the SFC with additional links according to re-order compatibility matrix such that all valid chains can be traced



Reorder(F1,F2) = true

If F2 is swapped with F1 then there can be a link from F1 to F3

ILP *selects the optimal set of virtual links* to form a valid SFC and performs joint NF placement and virtual link routing while minimizing network bandwidth usage

* Details are in the paper

OPT-Khaleesi*

Decision Variables

VNF Placement on Servers, Virtual Link Selection, Virtual Link Routing

Constraints

- Select exactly F - 1 virtual links (F = number of NFs)
- Selected virtual links should form a chain semantically equivalent to the input

* Details are in the paper

FAST-Khaleesi: 4-Step Algorithm

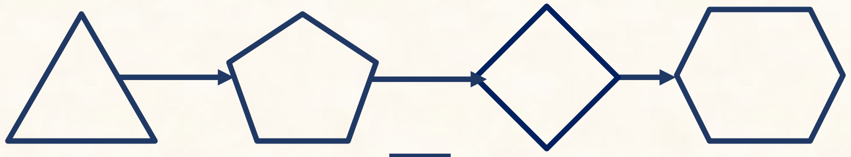
Step-I: Generate all possible SFCs

Step-II: Find Candidate Servers

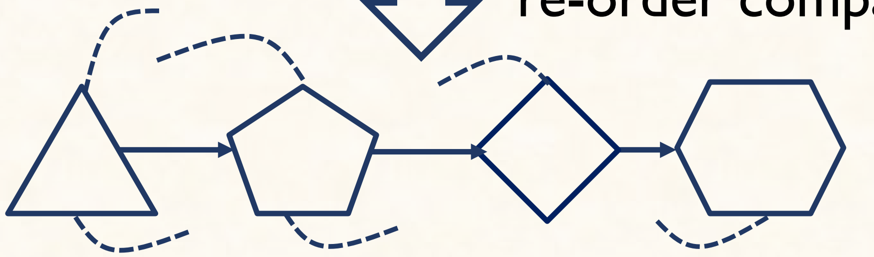
Step-III: Place VNFs on servers to minimize cross-server traffic

Step-IV: Inter-server Routing

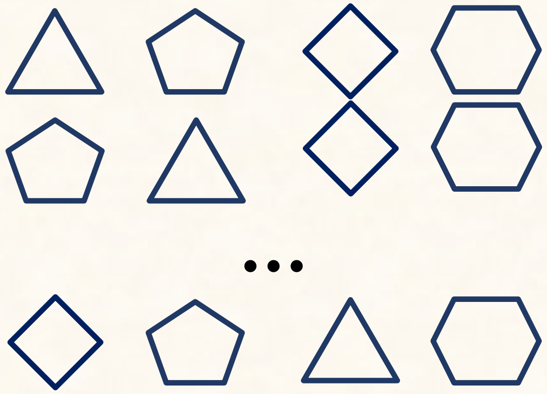
Step – I: SFC Generation



Augment edges according to re-order compatibility matrix



DFS Traversal to generate all chains



Step – II: Determine Candidate Servers

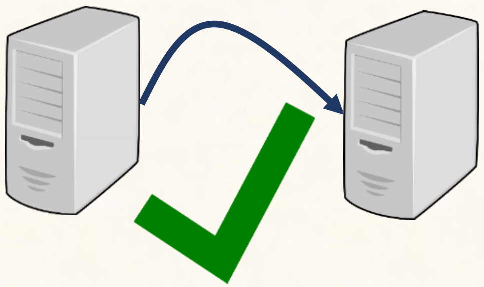
II.1 Filter servers based on NF's CPU requirement

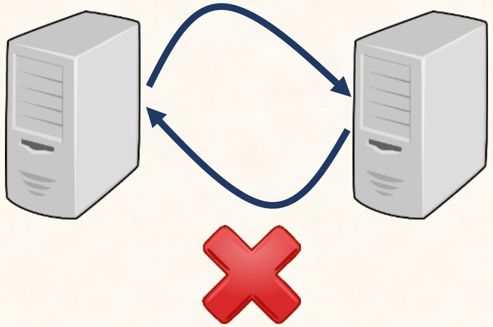
II.2 Sort servers in decreasing order of *R*

$$R = \frac{\text{Residual number of cores}}{\text{Distance from ingress sw + Distance from egress sw}}$$

Prioritize servers with higher capacity and proximity to the SFC's ingress and egress nodes

Step – III, IV: VNF Placement & Routing





III.1 Traverse servers in decreasing order of R

III.2 Keep placing VNFs from the beginning of a chain using first fit algorithm

III.3 Determine the number of cross server virtual links

III.4 Pick placement with minimum number of cross-server link

IV. Route virtual links between servers using Dijkstra's algorithm

Evaluation: Setup

- Comparison scenarios
  - OPT-Khaleesi compared with *optimal rigid SFC orchestration**
  - *An existing heuristic** for rigid SFC orchestration fed with all possible chains
  - FAST-Khaleesi compared with OPT-Khaleesi
- Substrate network
  - Small size campus data center (23 nodes, 42 links)
  - Moderate size ISP network from RocketFuel (79 nodes, 147 links)
- SFC request
  - 3 – 6 NFs/request
  - Poisson process (4 – 10 SFC/100 time unit avg., 1000 unit avg. lifetime)
  - Both real and synthetic traffic matrix

* Bari, *et al.* "Orchestrating Virtualized Network Functions ", IEEE TNSM 13(4): 913-926, 2016.

Performance Highlights

☺

Heuristic's acceptance Ratio within 20% of optimal on avg.

* Proof is in the paper

Performance Highlights

☺

Heuristic's acceptance Ratio within 20% of optimal on avg.

😐

~10% improvement in revenue/unit cost

Existing heuristic improves ~10% on revenue/cost without explicitly considering flexibility

* Proof is in the paper

Summary

We make a case for having flexibility in SFC traversal order

Two Solutions to Flexible SFC Orchestration: Khaleesi, FAST-Khaleeis

Presented quantifiable benefits of flexible SFC orchestration

What's Next?

Can we exploit re-order compatibility during failure restoration/re-optimization?

What is the impact on debugging and verification?

Can we automate middlebox characterization?

Backup Slide

FAST-Khaleesi: Complexity

$$O(N^2 lgN + SN'F^2 + S(L + NlgN)))$$

N = Number of switches

N'= Number of Servers

F = Number of NFs

S = Number of possible chains
(O(F!) in the worst case)