MŰEGYETEM 1782

Lendület
MTA-BME

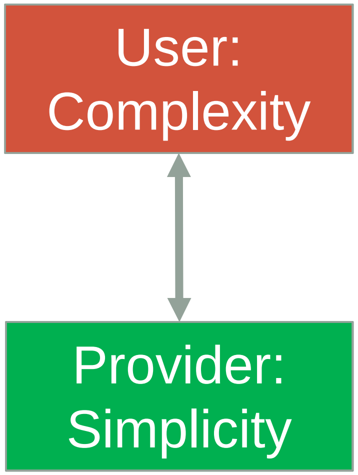
SURVIVABLE ROUTING ARCHITECTURES

Dr. Péter Babarczi
Assistant Professor

Budapest University of Technology and Economics
MTA-BME Lendület Future Internet Research Group

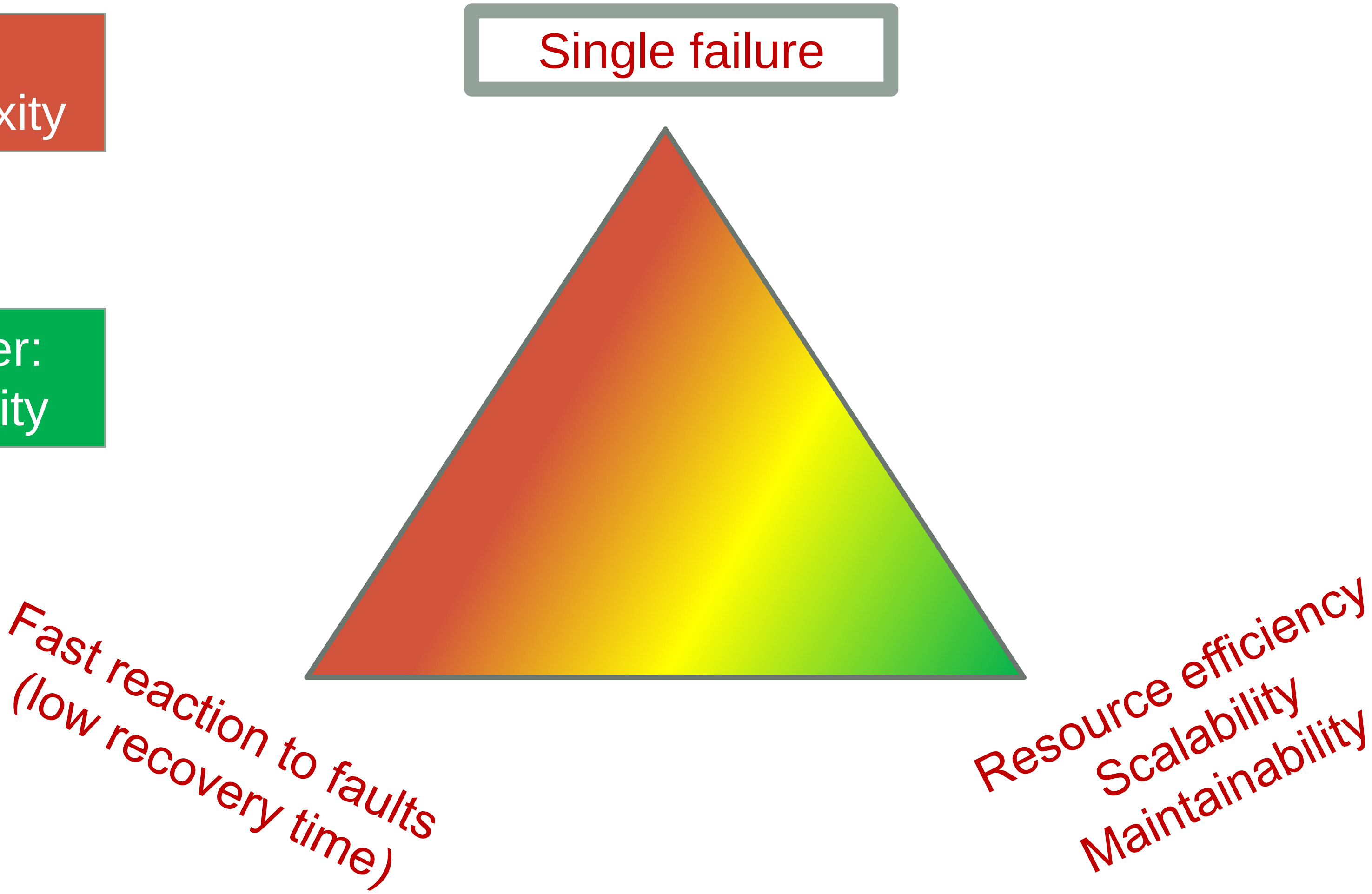
Survivable routing design

Contradicting requirements



User: Complexity

Provider: Simplicity



Single failure

Fast reaction to faults (low recovery time)

Resource efficiency
Scalability
Maintainability

Resource efficiency – recovery time trade-off

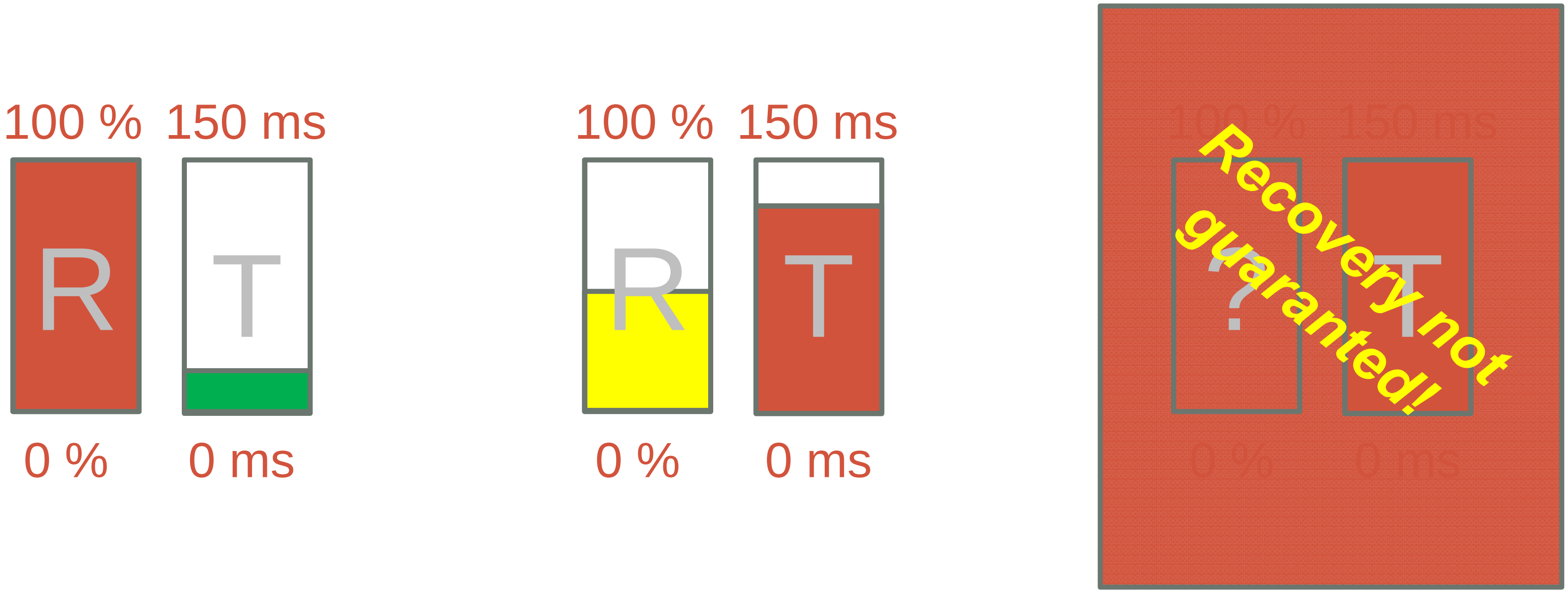
While a pre-defined availability is guaranteed

- Our goal: eliminate, or at least reduce the trade-off with state-of-the-art network coding and failure localization techniques.

Dedicated protection

Pre-planned restoration (Shared protection)

Dynamic restoration



100 % 150 ms R T 0 % 0 ms

100 % 150 ms R T 0 % 0 ms

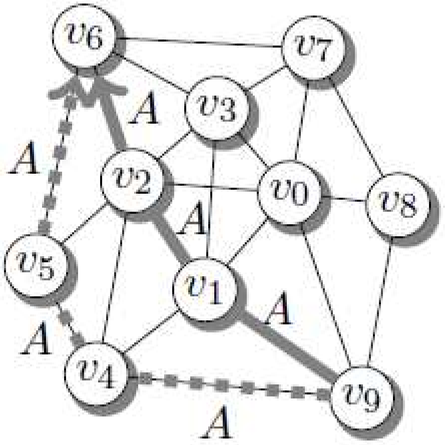
100 % 150 ms ? T 0 % 0 ms

Recovery not guaranteed!

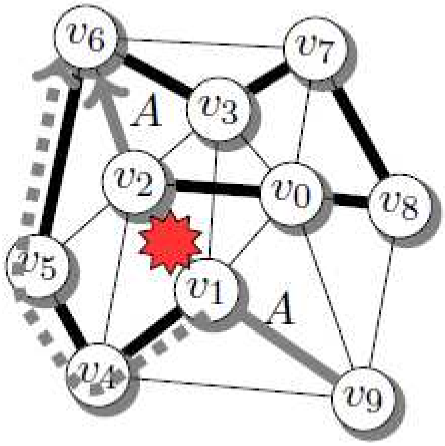
Benchmark dedicated/pre-planned protection methods

Data is sent on the working path (***W-LP***), protection path (***P-LP***) is reserved / configured

- ***1 + 1 protection** – **a survivable routing***
  - The same copy of user data *A* is sent along two disjoint paths between source node $v9$ and destination node $v6$.
  - Upon failure, only switching at the destination node required (simplest method)
  - No control plane signaling (quasi ***instantaneous recovery***)
  - **BUT:** ***More than 100% redundancy***



- ***P-Cycles*** – pre-planned but shared P-LP
  - W-LP $v_9 \rightarrow v_1 \rightarrow v_2 \rightarrow v_6$ is rerouted along the P-LP $v_9 \rightarrow v_1 \rightarrow v_4 \rightarrow v_5 \rightarrow v_6$ upon the on-cycle link failure of (v_1, v_2)
  - ***Resource efficient*** (share P-LP among multiple W-LPs): 60-70% redundancy
  - **BUT:** ***recovery is slow owing to rerouting*** (requires failure localization and switching matrix configuration of 40-50 ms)



SURVIVABLE ROUTING MEETS NETWORK CODING

- Static/dynamic dedicated protection approaches
- Instantaneous recovery with network coding
- MINERVA – Implementing network coding in SDNs

Survivable routing – Network model

Find a minimum cost (single link failure resilient) routing

- Transport network represented by a directed graph $\boldsymbol{G=(V,E,c,k)}$ with node set $\boldsymbol{V}$ and link set $\boldsymbol{E}$,
  - Free capacity $\boldsymbol{k(e)}$, i.e., number of available bandwidth units,
  - Cost $\boldsymbol{c(e)}$, i.e., cost of using one unit of bandwidth along link $\boldsymbol{e}$.
- Given a connection request $\boldsymbol{D=(s,t,d)}$,
  - with information source $\boldsymbol{s}$,
  - with information sink $\boldsymbol{t}$,
  - and the number of bandwidth units $\boldsymbol{d}$ requested for data transmission.
- Output:
  - minimum cost survivable routing in terms of cost function $\boldsymbol{c(e)}$.

  - **Definition** We say that $\boldsymbol{R=(VR,ER,f)}$ is a ***survivable routing*** of a connection $D=(s,t,d)$ in $\boldsymbol{G}$ with $\forall \boldsymbol{e} \in \boldsymbol{ER}$: $\boldsymbol{f(e) \leq k(e)}$, if there is an $\boldsymbol{s-t}$ flow $\boldsymbol{f}$ of value $\boldsymbol{F \geq d}$ in $\mathbf{R}$, even if we delete any single link of $\mathbf{R}$. On the other hand, a routing is *vulnerable* if it is not survivable.

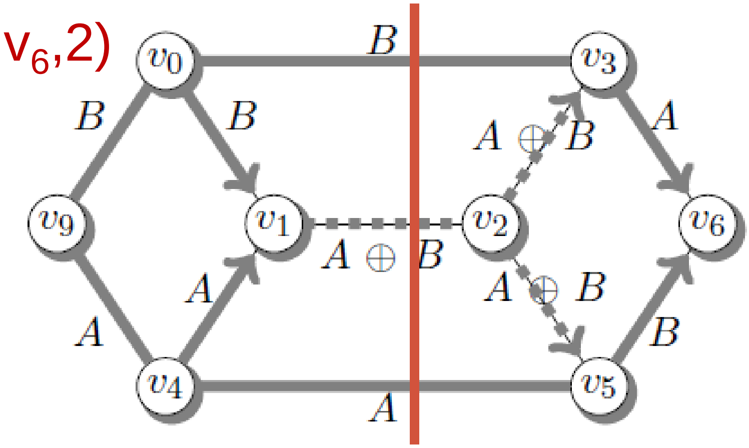
- GOAL: ***no flow rerouting or packet retransmission upon failure***

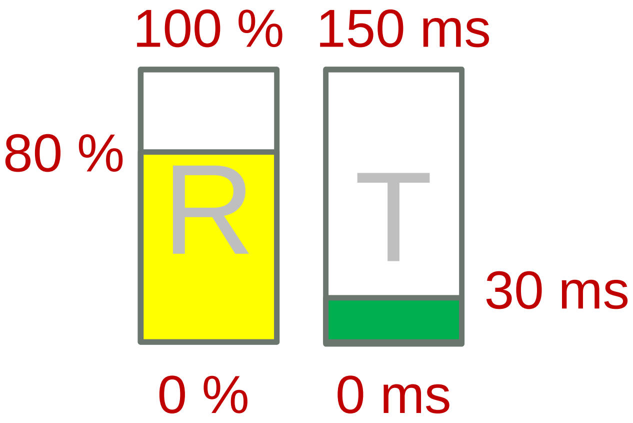
Network coding

Intermediate nodes are allowed to perform algebraic operations on the packets

- After the (link) failure is identified any routing method could be ***adopted and resend the flows*** on the intact edges of a survivable routing **R**, clearly resulting in ***slow recovery***.
- We have to break with the traditional store-and-forward networking concept to deploy optimal survivable routings
  - **(Network) coding is required to ensure *instantaneous recovery***

D=(v_9,v_6,2)

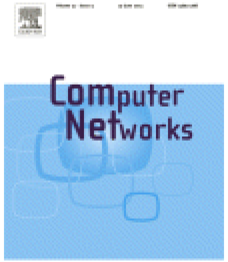




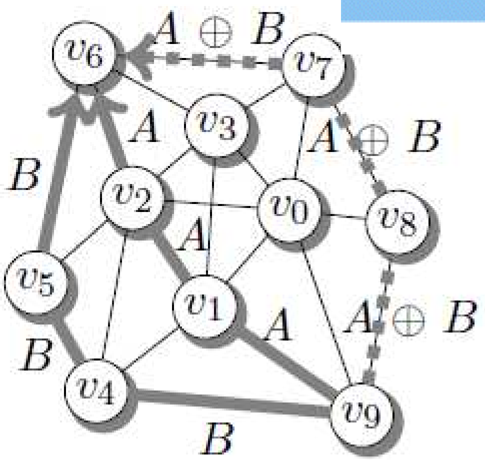
Péter Babarczi, János Tapolcai, Pin-Han Ho, and Muriel Médard, *Optimal Dedicated Protection Approach to Shared Risk Link Group Failures using Network Coding*, in Proceedings of the **IEEE International Conference on Communications (ICC)**, pp. 3051-3055, Ottawa, ON, Canada, 2013.

Dedicated protection - static routing

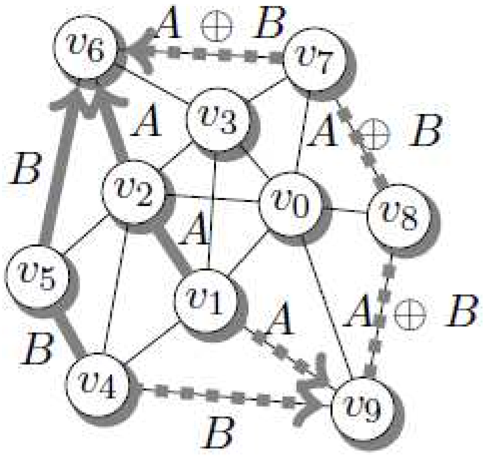
Improve resource efficiency with coding while **instantaneous recovery** is maintained



- ***Diversity coding***
  - The user data is split into two parts (*A* and *B*) and sent along three disjoint paths by adding redundancy *A XOR B*.
  - ***Good resource-efficiency*** (50% instead of 100% of 1+1)
  - **BUT**: requires 3-edge-connectivity



- ***Inter-session 1+1 network coding***
  - The data of connections v_4 to v_6 and from v_1 to v_6 is sent directly to the ***common destination*** node and to a ***coding node*** (v_9).
  - The most resource-efficient against *single link failures* (in static routing!)
  - **BUT:** ***NP-complete***

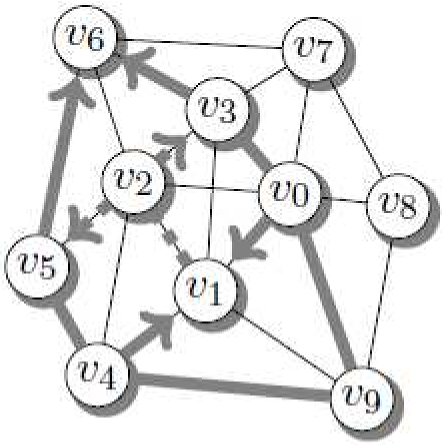


Péter Babarczi, Gergely Biczók, Harald Øverby, János Tapolcai, and Péter Soproni, *Realization Strategies of Dedicated Path Protection: A Bandwidth Cost Perspective*, **Elsevier Computer Networks (ComNet)**, vol. 57, no. 9, pp. 1974-1990, impact factor 1.282, 2013.

Dedicated protection - dynamic routing

Survivable routing methods with instantaneous recovery against multiple (shared risk link group) failures

- General Dedicated Protection (GDP)
  - Survivable routing ***does NOT have a pre-defined structure*** (e.g., disjoint path-pair)
- GDP with routing (GDP-R)
  - Traditional store-and-forward behavior
  - ***NP-complete for SRLG failures***
- GDP with network coding (GDP-NC)
  - *Intra-session network coding* is allowed
  - Minimal resource consumption among all methods ensuring instantaneous recovery (***polynomial-time***)
  - **BUT:** *User data might be split into arbitrary many parts*
    - Practically infeasible, but provides the ***theoretical lower bound***

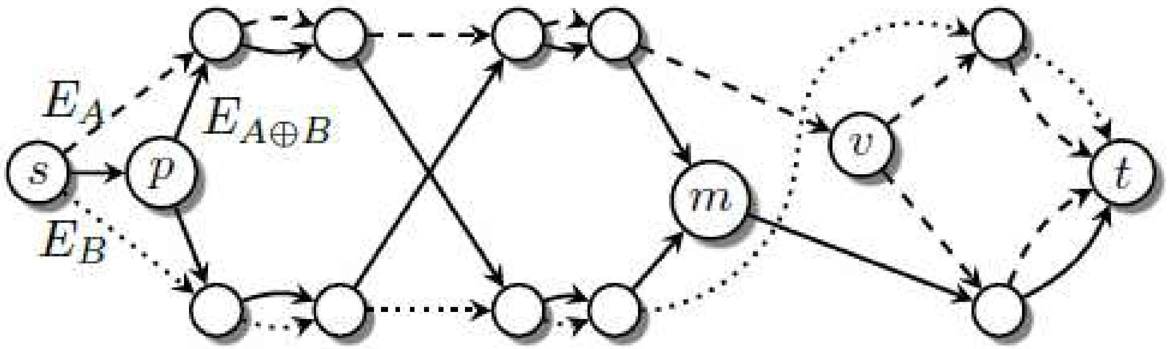


Péter Babarczi, Alija Pašić, János Tapolcai, Felicián Németh, and Bence Ladóczki, *Instantaneous Recovery of Unicast Connections in Transport Networks: Routing versus Coding,* accepted to **Elsevier Computer Networks (ComNet)**, impact factor 1.282, 2015.

Resilient Flow Decomposition (RFD)

A practical special case: single link failure resilience with two data parts

- **Theorem**: Suppose that survivable routing is ***critical***. Then there are disjoint edge sets of , called ***routing DAGs***, such that for an arbitrary edge after removing the corresponding edge(s) from at least two of the routing DAGs connect to .
  - Only splitters (***p***) and mergers (***m***) are required at the intermediate nodes, ***XOR coding is sufficient at the source and destination nodes***.
  - Network code (i.e., routing DAGs) can be found in ***linear time*** in a critical survivable routing.
  - A critical survivable routing can be found in ***polynomial time***.

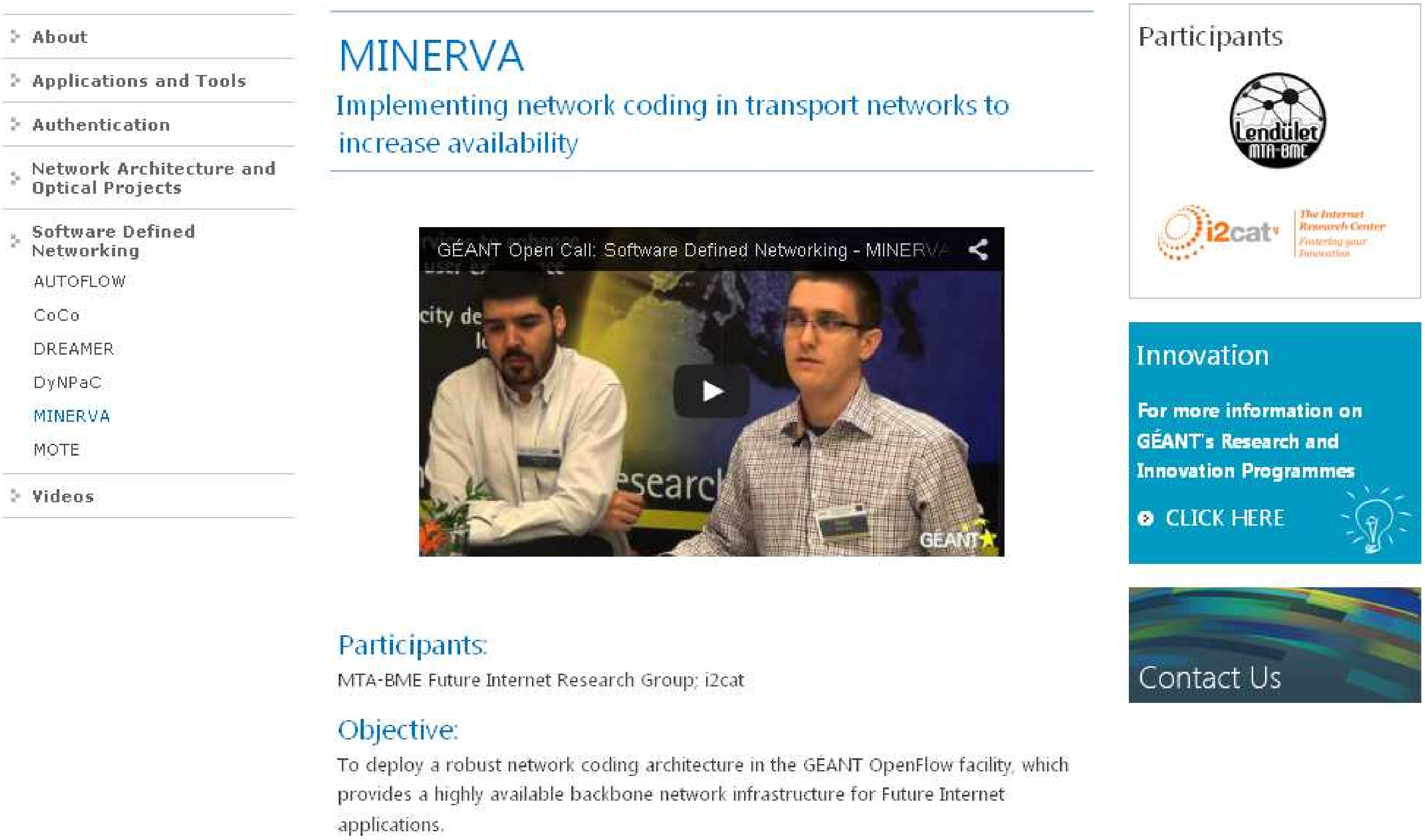


Péter Babarczi, János Tapolcai, Lajos Rónyai, and Muriel Médard, *Resilient Flow Decomposition of Unicast Connections with Network Coding*, in Proceedings of the **IEEE International Symposium on Information Theory (ISIT)**, pp. 116-120, Honolulu, HI, USA, 2014.

MINERVA

Open Call beneficiary of the GN3Plus (GÉANT) FP7 Project (2013 November – 2015 March)

- Implementing RFD in the GÉANT OpenFlow Facility (GOFF)
- http://www.geant.net/opencall/SDN/Pages/MINERVA.aspx


About
Applications and Tools
Authentication
Network Architecture and Optical Projects
Software Defined Networking
AUTOFLOW
CoCo
DREAMER
DyNPaC
MINERVA
MOTE
Videos
GÉANT Open Call: Software Defined Networking - MINERVA
Participants
Lendület MTA-BME
i2cat The Internet Research Center Fostering your Innovation
Innovation
For more information on GÉANT's Research and Innovation Programmes
CLICK HERE
Contact Us

MINERVA

Implementing network coding in transport networks to increase availability

Participants:

MTA-BME Future Internet Research Group; i2cat

Objective:

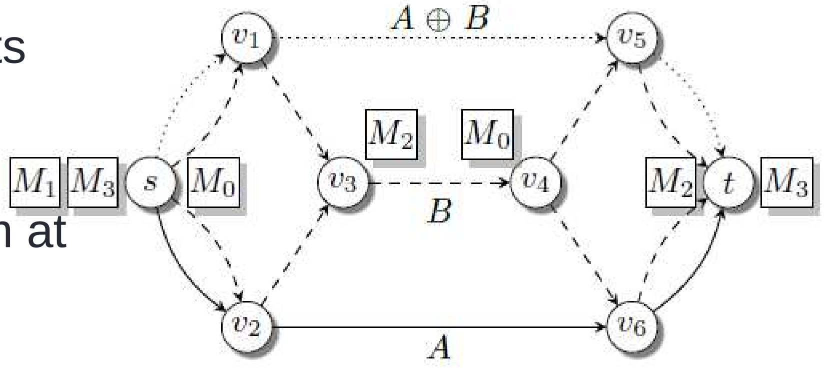
To deploy a robust network coding architecture in the GÉANT OpenFlow facility, which provides a highly available backbone network infrastructure for Future Internet applications.

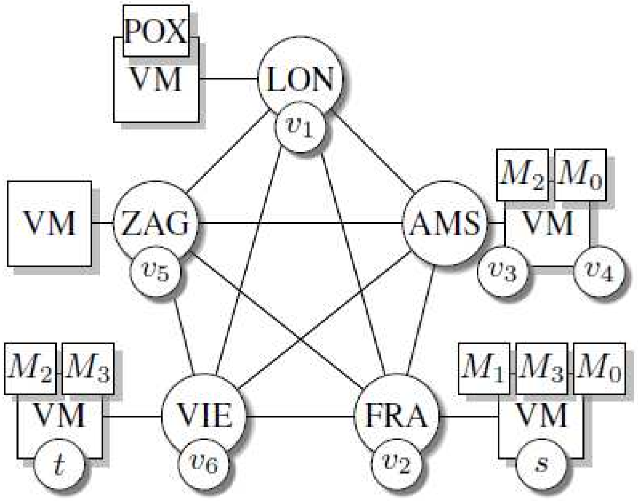
IEEE INFOCOM'15
26 APRIL - 1 MAY, 2015 • HONG KONG
SHERATON HONG KONG HOTEL AND TOWERS

Virtual Network Functions

An arbitrary RFD solution can be deployed with these NFs

- ***Splitter (M_0)***: duplicates incoming packets and forwards them through two different links (s and v_4).
- ***Sequencer (M_1)***: divides the input stream at the source node s into two parts *A* and *B* (e.g., based on parity).
- ***Merger (M_2)***: receives the same flow on two incoming links and forwards one of them (or the intact one upon link failure) through its single outgoing link (v_3 and t).
- ***Coding/Decoding (M_3)***: perform fast packet processing using XOR operation and queues to handle the incoming packets (they are always placed at s and t).





Bence Ladóczki, Carolina Fernandez, Oscar Moya, Péter Babarczi, János Tapolcai, and Daniel Guija, *Robust Network Coding in Transport Networks*, in Proceedings of the **34th IEEE International Conference on Computer Communications (INFOCOM) Demo session**, pp. 1-2, Hong Kong, 2015.

ALL-OPTICAL FAILURE LOCALIZATION

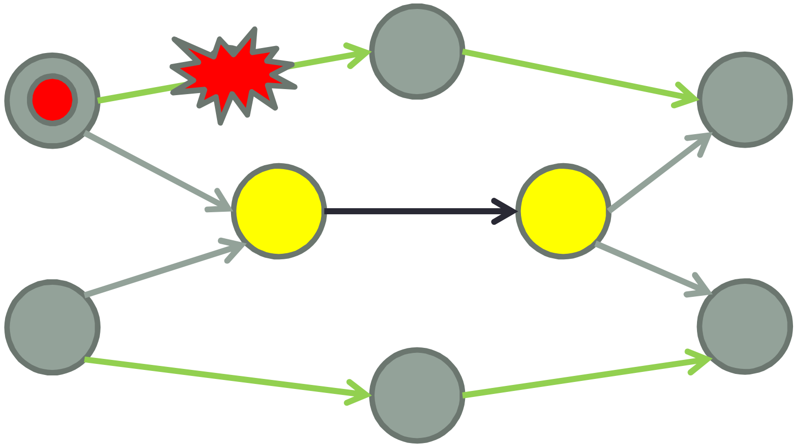
- GMPLS-based failure recovery (identifying week points)
- Failure localization via a central failure manger
- Distributed failure localization (enabling shared protection schemes in practice)

Pre-planned restoration (shared protection)

GMPLS-based failure recovery

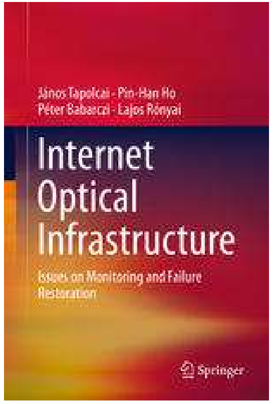
- Real-time operations after a failure (recovery time (***t_R***))
  - Failure detection
  - Failure localization (***t_l***)
  - Failure notification (***t_n***)
  - Failure correlation (***t_c***)
  - Failure restoration
    - Path selection (***t_p***)
    - Device configuration (***t_d***)

Failure management



t_R at shared protection (150 ms):

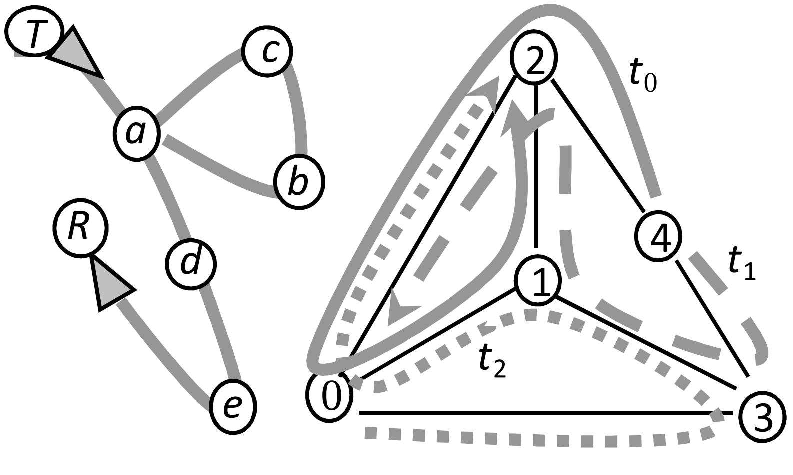
- ***t_l*** = 10 ms
- ***t_n*** = 20-30 ms
- ***t_c*** = 20-30 ms
- ***t_p*** = 0-30 ms
- ***t_d*** = 50 ms


János Tapolcai · Pin-Han Ho
Péter Babarczi · Lajos Rónyai
Internet Optical Infrastructure
Issues on Monitoring and Failure Restoration
Springer

All-optical failure localization

Introducing supervisory lightpaths (S-LPs) or m-trails

- ***Unambiguous Failure Localization***
  - Each link is traversed by a unique set of m-trails or equivalently each row of the alarm code table is unique



(a) m-trail

(b) An m-trail solution

Link	t_2	t_1	t_0	Decimal
(0,1)	1	0	1	5
(0,2)	1	1	1	7
(0,3)	1	0	0	4
(1,2)	0	1	1	3
(1,3)	1	1	0	6
(2,4)	0	0	1	1
(3,4)	0	1	0	2

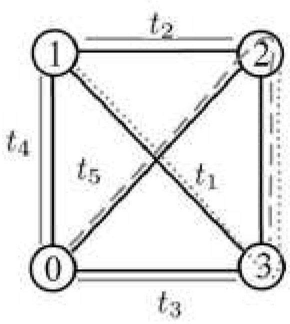
(c) Alarm code table

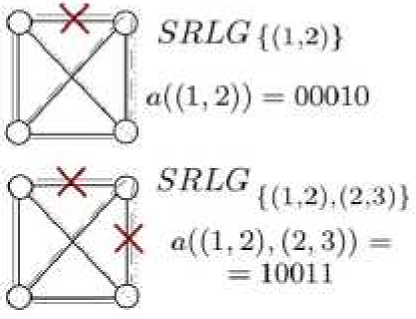
János Tapolcai, Pin-Han Ho, Péter Babarczi, and Lajos Rónyai, *Internet Optical Infrastructure - Issues on Monitoring and Failure Restoration*, pp. 1-212, **Publisher: Springer**, ISBN 978-1-4614-7737-2, 2014.

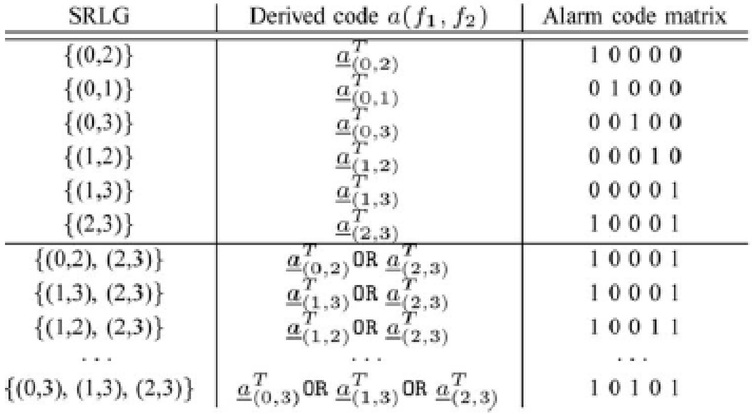
All-optical failure localization

Via a central failure manager

- Optimization goal:
  - ***Minimize the number of S-LPs*** (alarms) causing failure localization complexity (i.e., increased recovery time)
- Fast heuristics for single link failure UFL
- Localizing multiple link (shared risk link group, SRLG) failures is ***NP-complete***
  - Owing to the dependency of the physical S-LPs and logical m-trails





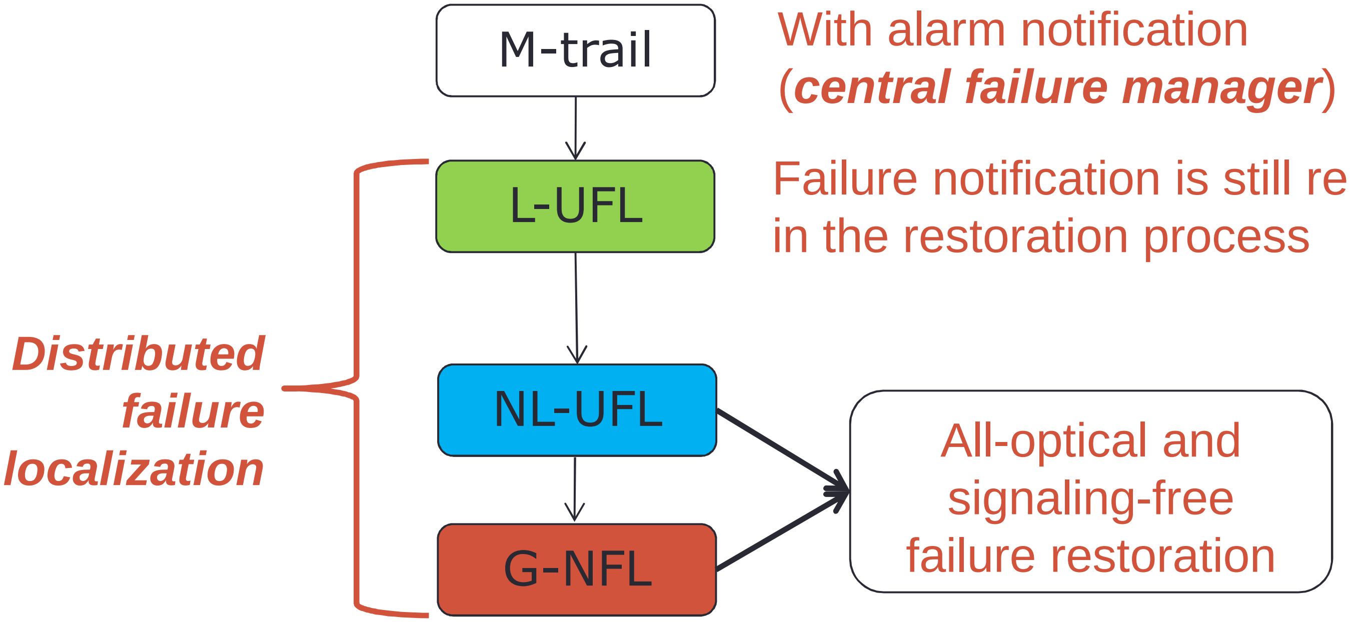


SRLG	Derived code $a(f_1, f_2)$	Alarm code matrix
$\{(0,2)\}$	$\underline{a}^T_{(0,2)}$	1 0 0 0 0
$\{(0,1)\}$	$\underline{a}^T_{(0,1)}$	0 1 0 0 0
$\{(0,3)\}$	$\underline{a}^T_{(0,3)}$	0 0 1 0 0
$\{(1,2)\}$	$\underline{a}^T_{(1,2)}$	0 0 0 1 0
$\{(1,3)\}$	$\underline{a}^T_{(1,3)}$	0 0 0 0 1
$\{(2,3)\}$	$\underline{a}^T_{(2,3)}$	1 0 0 0 1
$\{(0,2), (2,3)\}$	$\underline{a}^T_{(0,2)}$ OR $\underline{a}^T_{(2,3)}$	1 0 0 0 1
$\{(1,3), (2,3)\}$	$\underline{a}^T_{(1,3)}$ OR $\underline{a}^T_{(2,3)}$	1 0 0 0 1
$\{(1,2), (2,3)\}$	$\underline{a}^T_{(1,2)}$ OR $\underline{a}^T_{(2,3)}$	1 0 0 1 1
. . .	. . .	. . .
$\{(0,3), (1,3), (2,3)\}$	$\underline{a}^T_{(0,3)}$ OR $\underline{a}^T_{(1,3)}$ OR $\underline{a}^T_{(2,3)}$	1 0 1 0 1

Péter Babarczi, János Tapolcai, and Pin-Han Ho, *Adjacent Link Failure Localization with Monitoring Trails in All-Optical Mesh Networks*, **IEEE/ACM Transactions on Networking (ToN)**, vol. 19, no. 3, pp. 907-920, impact factor 2.033, 2011.

All-optical failure localization

Using S-LPs in failure restoration (e.g., Failure Dependent Protection, FDP)


M-trail
L-UFL
NL-UFL
G-NFL
Distributed failure localization
All-optical and signaling-free failure restoration

With alarm notification (***central failure manager***)

Failure notification is still required in the restoration process

- ***L-UFL:*** Local UFL – a single node performs UFL based on the status of the traversing S-LPs
- ***NL-UFL:*** Network-wide L-UFL – each node in the network is L-UFL capable
- ***G-NFL:*** neighborhood failure localization – localize only links relevant for FDP

János Tapolcai, Pin-Han Ho, Péter Babarczi, and Lajos Rónyai, *On Signaling-Free Failure Dependent Restoration in All-Optical Mesh Networks*, **IEEE/ACM Transactions on Networking (ToN)**, vol. 22, no. 4, pp. 1067-1078, impact factor 1.986, 2014.

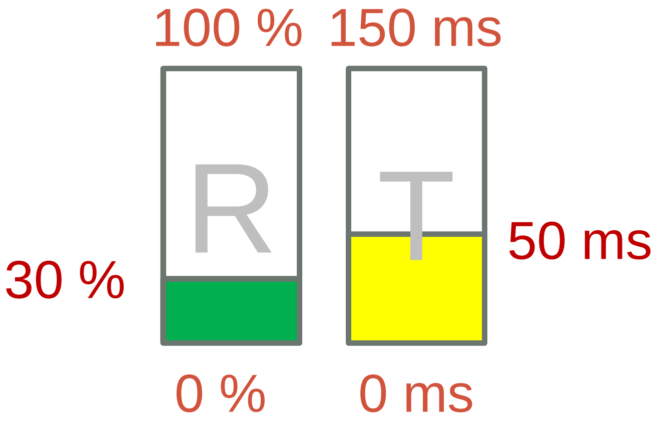
IEEE INFOCOM

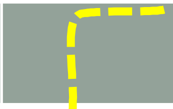
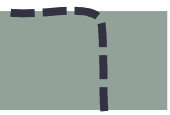
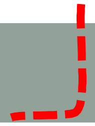
April 14-19, 2013 - Turin, Italy

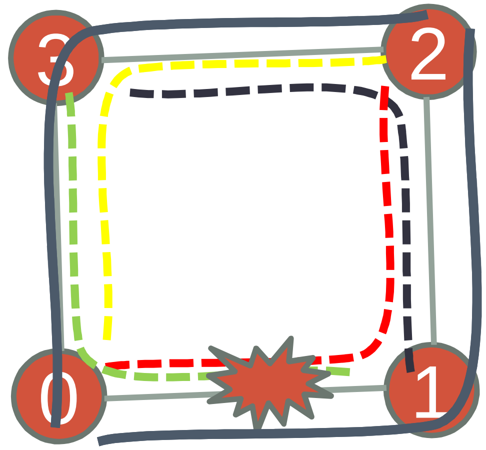
All-optical failure localization

Completely eliminating control plane signaling

- Each (switching) node can recover the disrupted connections ***without any control plane signaling***
  - After the failure localized, the node can immediately start switching matrix configuration (sub 50 ms recovery)
  - The number of alarms is no longer a concern (goal: ***minimize the total length of m-trails***)



Node 2				Action
0-1	0	0	1	SW 1->3
1-2	0	1	1	
2-3	1	1	0	
3-0	1	0	0	



János Tapolcai, Pin-Han Ho, Péter Babarczi, and Lajos Rónyai, *On Achieving All-Optical Failure Restoration via Monitoring Trails*, in Proceedings of the **32nd IEEE International Conference on Computer Communications (INFOCOM),** pp. 380-384, Turin, Italy, 2013.

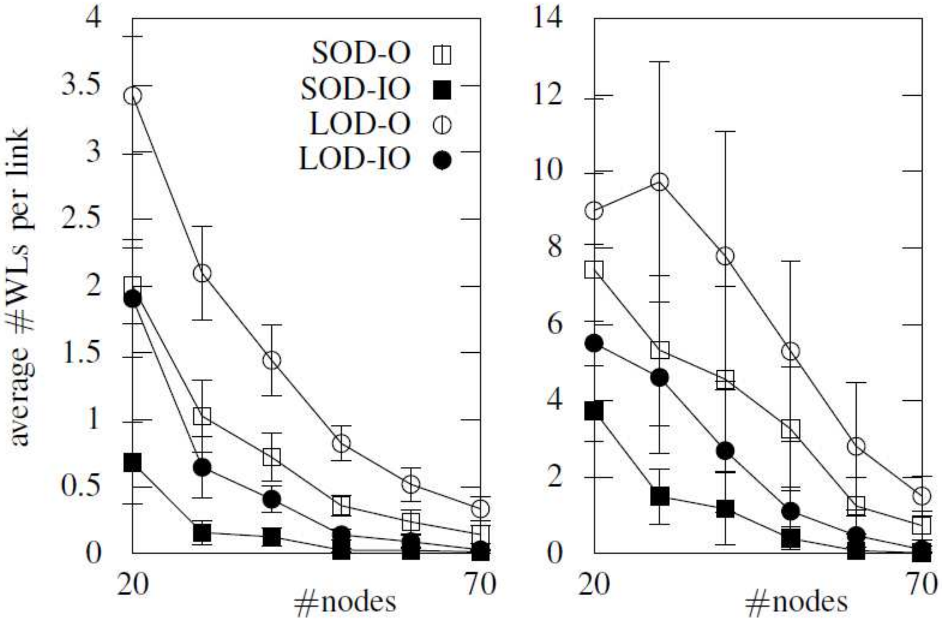
All-optical failure localization

Protection resources „hide" the S-LP capacity

- What is the price we have to pay for fast recovery?
  - The additional S-LP capacity is negligible even in lightly loaded networks

W-LP is established between 30% of *s-t* paris

-O: only S-LPs are tapped
-IO: data plane (W-LPs) are tapped as well -> reconfiguration



(a) Dense networks (b) Sparse networks

Fig. 6. Monitoring overhead that cannot be hidden by the spare capacity.

János Tapolcai, Pin-Han Ho, Péter Babarczi, and Lajos Rónyai, *Neighborhood Failure Localization in All-Optical Networks via Monitoring Trails*, accepted to **IEEE/ACM Transactions on Networking (ToN)**, impact factor 1.986, 2015.

MTA-BME FUTURE INTERNET RESEARCH GROUP

- Internet routing – Compressing IP forwarding tables
- Bloom filter based future Internet addressing
- ESCAPE - SDN prototyping framework

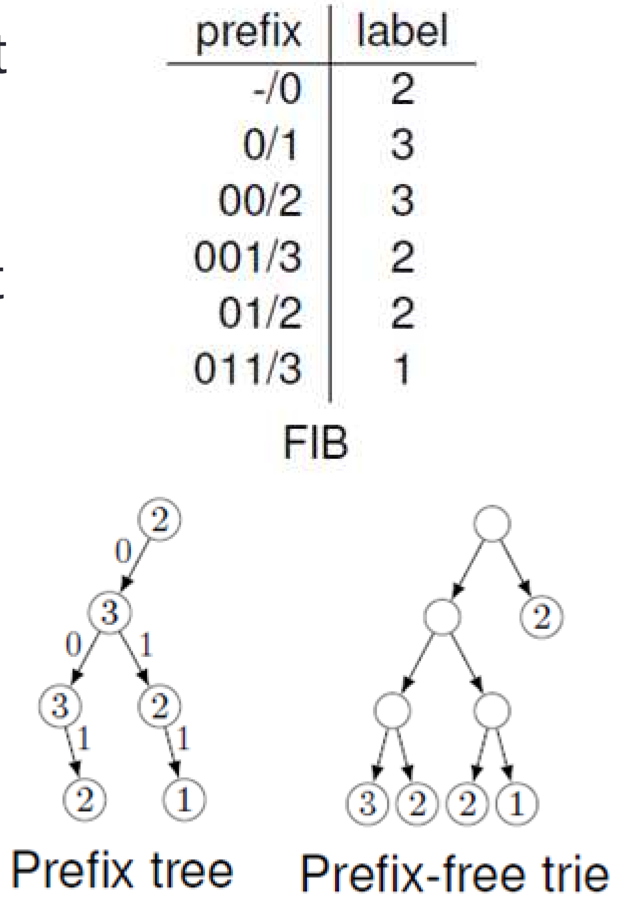
Group leader: Dr. János Tapolcai http://lendulet.tmit.bme.hu/

Compressing IP forwarding tables

Compressed data structures

SIGCOMM 2013 Hong Kong

- Compression not necessarily sacrifices fast access!
- Store information in entropy-bounded space and provide fast in-place access to it
  - take advantage of regularity, if any, to ***compress data drifts closer to the CPU in the cache hierarchy*** operations are even faster than on the original uncompressed form
- No space-time trade-off!
- Goal: advocate compressed data structures to the networking community
- ***IP forwarding table compression as a use case***



prefix	label
-/0	2
0/1	3
00/2	3
001/3	2
01/2	2
011/3	1

FIB

Prefix tree

Prefix-free trie

Gábor Rétvári, János Tapolcai, Attila Kőrösi, András Majdán, and Zalán Heszberger, *Compressing IP forwarding tables: towards entropy bounds and beyond*. In *Proceedings of the* **ACM SIGCOMM,** pp. 111-122, ACM, New York, NY, USA, 2013.

Compressing IP forwarding tables

IP Forwarding Information Base

- The fundamental data structure used by IP routers to make forwarding decisions
- Stores more than 440K IP-prefix-to-nexthop mappings as of January, 2013
  - consulted on a packet-by-packet basis at line speed
  - queries are complex: longest prefix match
  - ***updated couple of hundred times per second***
  - takes several MBytes of fast line card memory and counting
- May or may not become an ***Internet scalability barrier***
- With the proposed compressed IP FIB in Linux kernel prototype
  - Several million lookups per sec both in HW and SW
  - faster than the uncompressed form
  - Size of 100-400 KB
  - tolerates more than 100, 000 updates per sec

Gábor Rétvári, János Tapolcai, Attila Kőrösi, András Majdán, and Zalán Heszberger, *Compressing IP forwarding tables: towards entropy bounds and beyond*. In *Proceedings of the* **ACM SIGCOMM,** pp. 111-122, ACM, New York, NY, USA, 2013.

Future Internet addressing and forwarding

Information Centric Networks (ICNs)

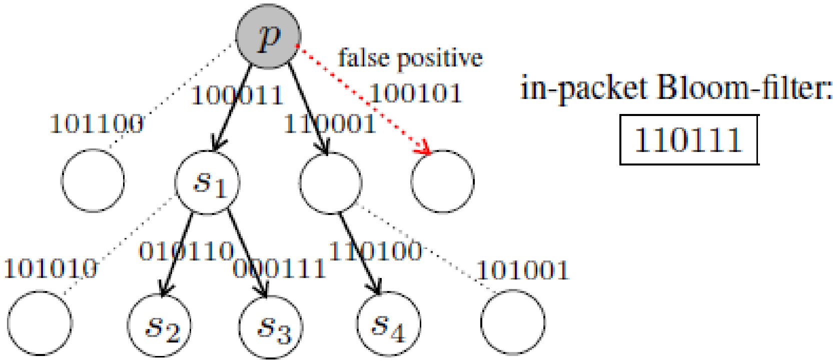
- Publish/subscribe service model
  - Spatially and Temporally Decouple Communicating Parties
    - producer of information (publisher) does not need to coexist in time with the consumers (subscribers)
  - Clearly Separate Network Functions
    - ***rendezvous*** matches demand for and supply of information
    - ***topology management*** and formation, to determine a suitable forwarding architecture (e.g., multicast tree)
    - this transfer being executed by the third function, ***forwarding***.
- The ICN paradigm shifts communication goal: ***what*** information required is more important than ***where*** it is in the network
  - Traditional IP forwarding mechanisms (based on end-point addresses) does not work (**architectural change required**)
  - Typically point-multipoint communication, but naturally support multipath routing for unicast connections as well.

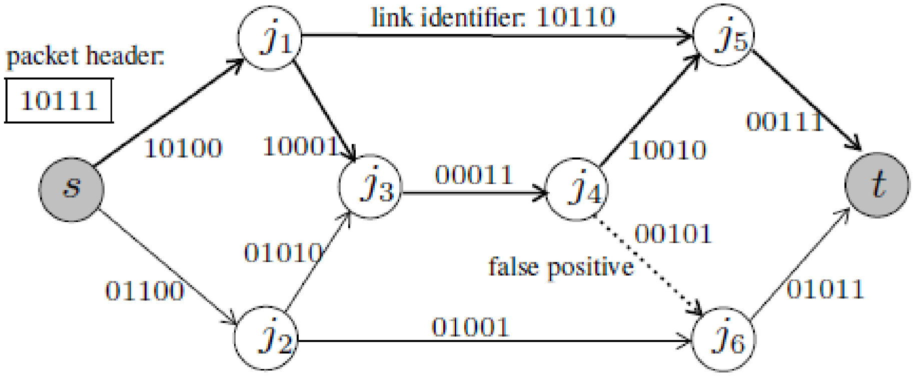
János Tapolcai, József Bíró, Péter Babarczi, András Gulyás, Zalán Heszberger, and Dirk Trossen, *Optimal False-Positive-Free Bloom Filter Design for Scalable Multicast Forwarding*, accepted to **IEEE/ACM Transactions on Networking (ToN)**, impact factor 1.986, 2015.

Future Internet addressing and forwarding

Forwarding based on in-packet Bloom filters

- Computation of in-packet Bloom filters
  - Each link is assigned by a binary link address (consisting of bits of which at most ***k*** are set to 1).
  - The topology manager computes the packet header by bitwise OR-ing the addresses of the links in the corresponding ***multicast tree / routing DAG***
  - ***SDN prototype*** implementation
  - In ToN: how to design „short" in-packet filters resulting (statistically) zero false-positive forwarding





János Tapolcai, József Bíró, Péter Babarczi, András Gulyás, Zalán Heszberger, and Dirk Trossen, *Optimal False-Positive-Free Bloom Filter Design for Scalable Multicast Forwarding*, accepted to **IEEE/ACM Transactions on Networking (ToN)**, impact factor 1.986, 2015.

ESCAPE

Extensible Service ChAin Prototyping Environment using Mininet, Click, NETCONF and POX

SIGCOMM CHICAGO 2014

- ***Service Chaining with SDN/NFV***
  - Dynamic service creation for infrastructure/service providers
  - Service is described as a graph of service components
  - Service components operates as Virtual Network Functions
    - simple packet manipulation: header rewrite, network coding
    - more complex tasks: Firewall, NAT
  - Network is programmed (SDN) to steer traffic to VNFs

- **SDN:** Remote Controller programs the forwarding behavior of switches
- **NFV:** Middleboxes run in virtual machines

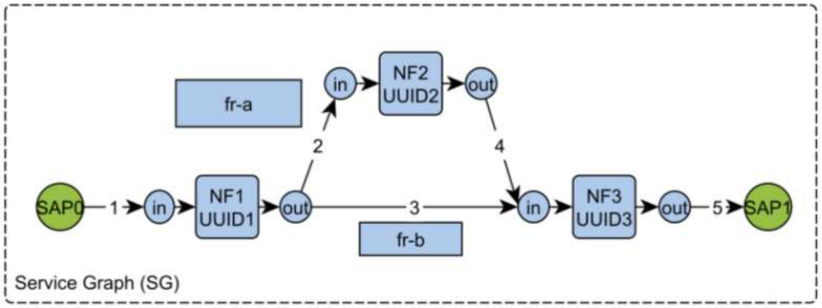


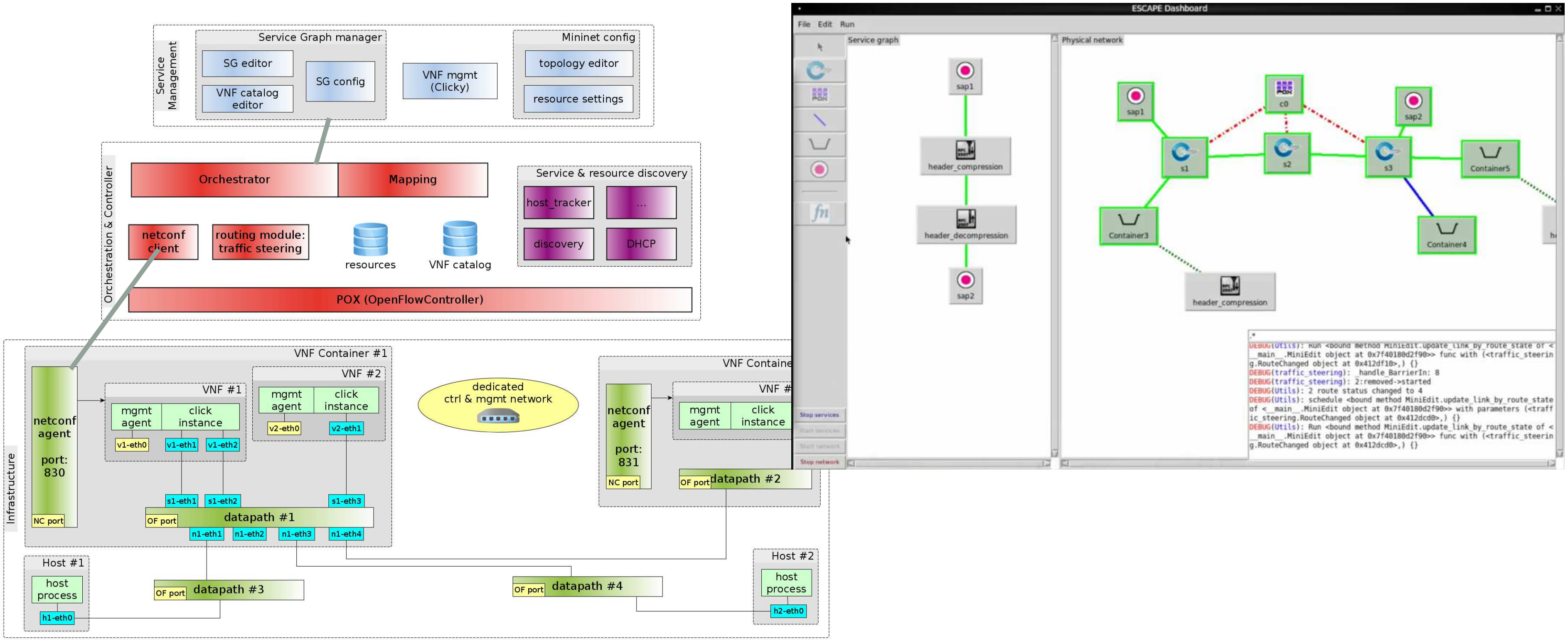
Figure 5: An illustrative example of a Service Graph (SG)

A. Csoma, B. Sonkoly, L. Csikor, F. Németh, A. Gulyás, W. Tavernier, and S. Sahhaf, *Escape: extensible service chain prototyping environment using mininet, click, netconf and pox*, in **ACM SIGCOMM (Demo)**, pp. 125-126. 2014.

ESCAPE

Extensible Service ChAin Prototyping Environment using Mininet, Click, NETCONF and POX

SIGCOMM CHICAGO 2014



```
.*
DEBUG(Utils): Run <bound method MiniEdit.update_link_by_route_state of <
__main__.MiniEdit object at 0x7f40180d2f90>> func with (<traffic_steerin
g.RouteChanged object at 0x412df10>,) {}
DEBUG(traffic_steering): _handle_BarrierIn: 8
DEBUG(traffic_steering): 2:removed->started
DEBUG(Utils): 2 route status changed to 4
DEBUG(Utils): schedule <bound method MiniEdit.update_link_by_route_state
of <__main__.MiniEdit object at 0x7f40180d2f90>> with parameters (<traff
ic_steering.RouteChanged object at 0x412dcd0>,) {}
DEBUG(Utils): Run <bound method MiniEdit.update_link_by_route_state of <
__main__.MiniEdit object at 0x7f40180d2f90>> func with (<traffic_steerin
g.RouteChanged object at 0x412dcd0>,) {}
```

https://sb.tmit.bme.hu/mediawiki/index.php/ESCAPE

A. Csoma, B. Sonkoly, L. Csikor, F. Németh, A. Gulyás, W. Tavernier, and S. Sahhaf, *Escape: extensible service chain prototyping environment using mininet, click, netconf and pox*, in **ACM SIGCOMM (Demo)**, pp. 125-126. 2014.

THANK YOU!

QUESTIONS?

Dr. Péter Babarczi babarczi@tmit.bme.hu