CT Scan for Your Network: Topology Inference from End-to-End Measurements

Ting He

Associate Professor, CSE@Penn State

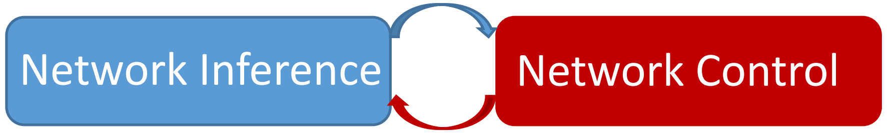
Students:

Yilei Lin, Yudi Huang, Akash Kumar (Penn State)

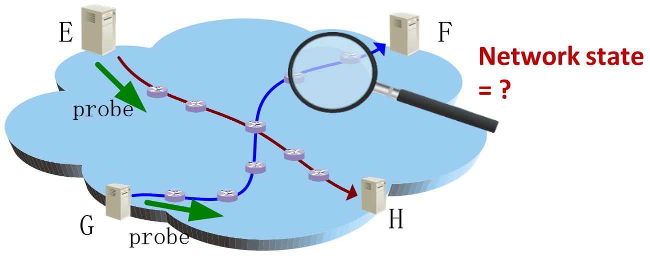
Network Sciences Research Group (NSRG)

- **Interests:**
  - **communication networking** (network tomography, SDN, overlay, 5G, security)
  - **distributed machine learning** (coreset, data reduction, federated learning)
  - **mobile edge computing** (resource allocation)
  - **cyber-physical systems** (smart grid, state estimation, false data injection)

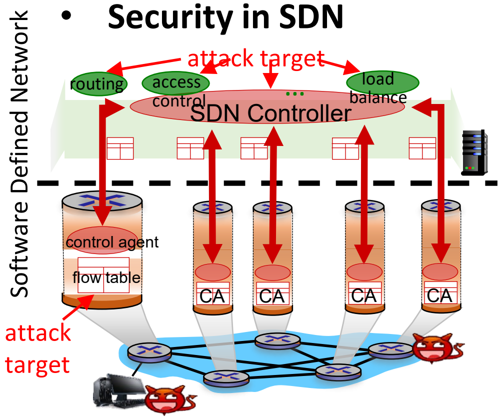
- **Members:**
  - **Ting He, Associate Professor**
  - 6 PhD students
  - Alumni: 4 PhD, 6 MS (Bucknell, Google, Meta, ByteDance, HP, Amazon, Oracle)


Network Inference
Network Control

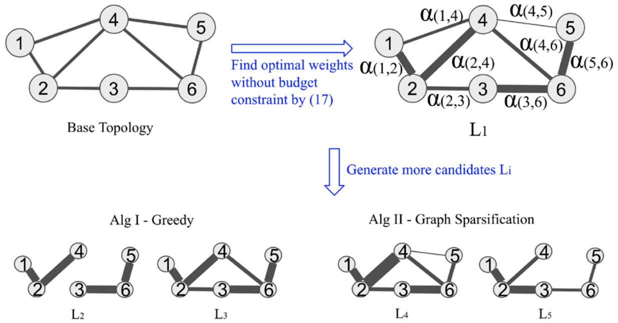
- **Example projects:**
  - **Network tomography**



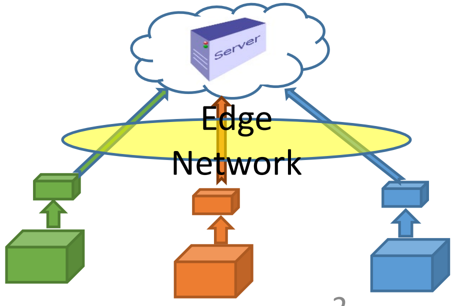
  - **Security in SDN**



  - **Communication-efficient ML**

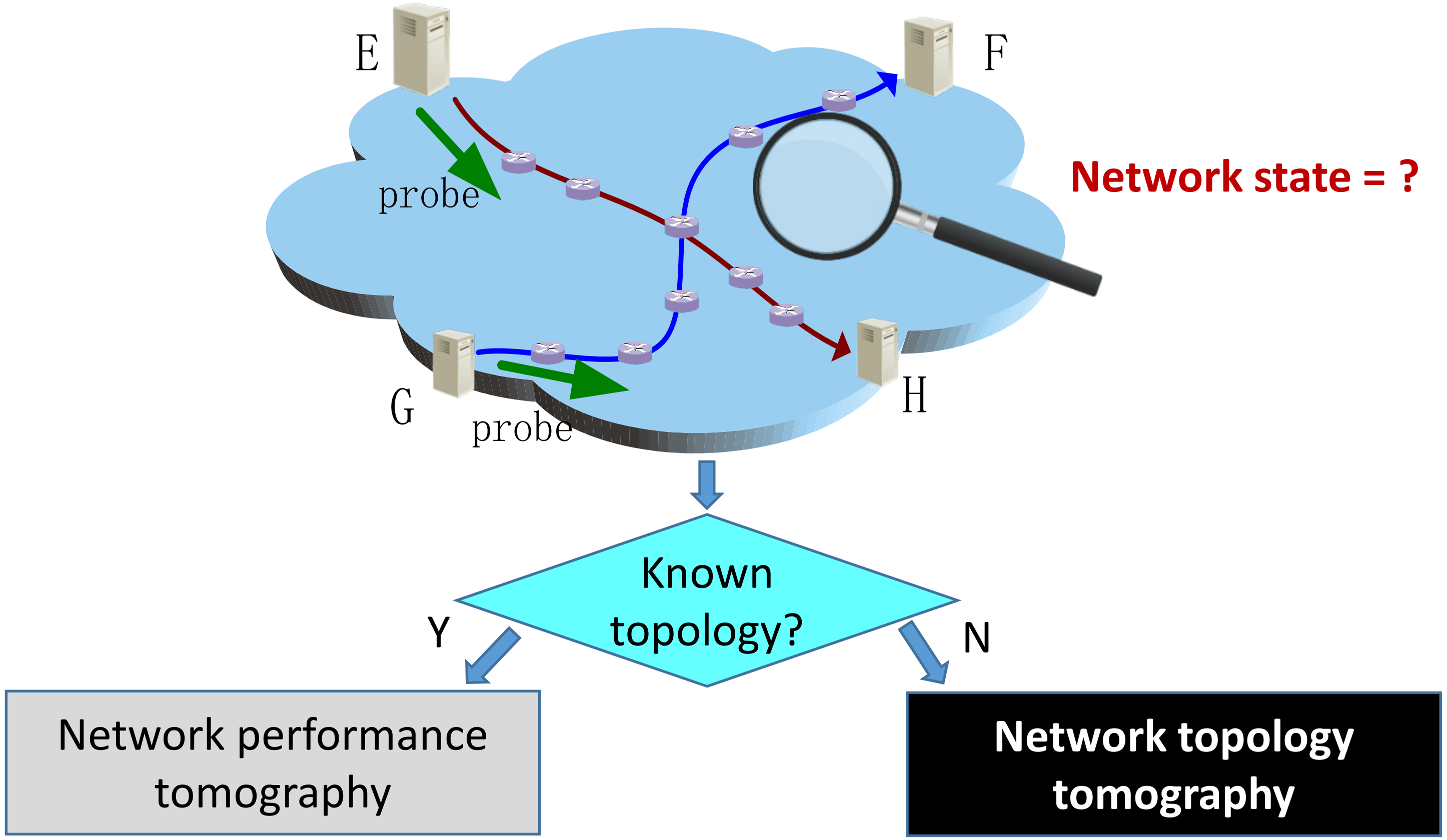

Base Topology
Find optimal weights without budget constraint by (17)
α(1,4)
α(4,5)
α(4,6)
α(1,2)
α(2,4)
α(5,6)
α(2,3)
α(3,6)
L1
Generate more candidates Li
Alg I - Greedy
Alg II - Graph Sparsification
L2
L3
L4
L5

  - **Data reduction for ML**



Overview: What is network tomography

- Using *external observations* to infer *internal network state*



Network state = ?

Known topology?

Y

N

Network performance tomography

Network topology tomography

A “CT scan” for the network!

Motivation: Why topology inference

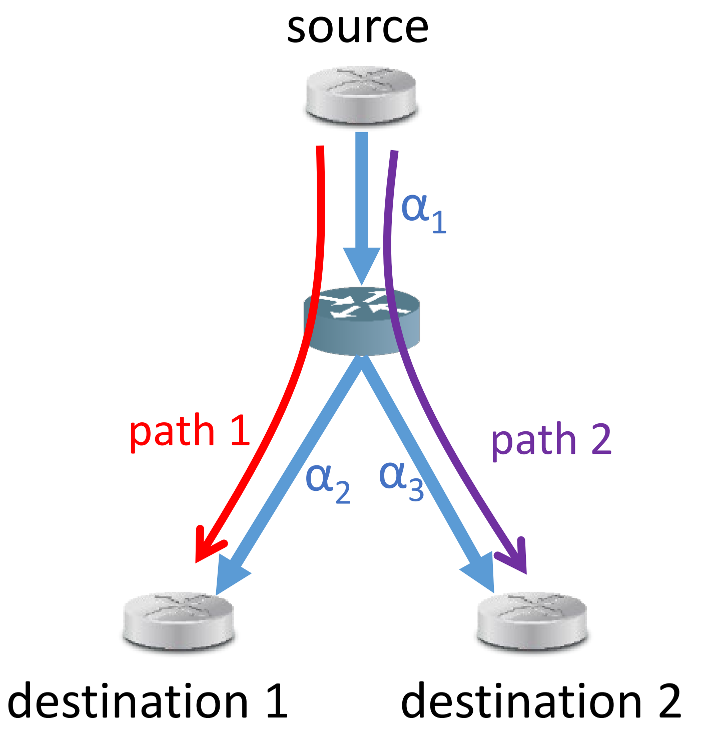
- **Topology information is useful**
  - Routing
  - Service placement
  - Client-server association
  - Overlay management
  - Load balancing
  - Trouble shooting
  - …

- But **it is not always observable**
  - Use protocols to collect topology information (e.g., SNMP, OpenFlow) → admin privilege
  - Use ICMP to measure topology (e.g., traceroute) → supportive internal nodes

Q: Is it possible to infer **network topology** from **end-to-end measurements**? If so, how?

Toy example: Why it is feasible

- Multicast measurements reveal internal topology



α_i success probability of a link

$$-\log\alpha_1 - \log\alpha_2 = -\log \Pr\{X_{p_1} = 1\},$$

$$-\log\alpha_1 - \log\alpha_3 = -\log \Pr\{X_{p_2} = 1\},$$

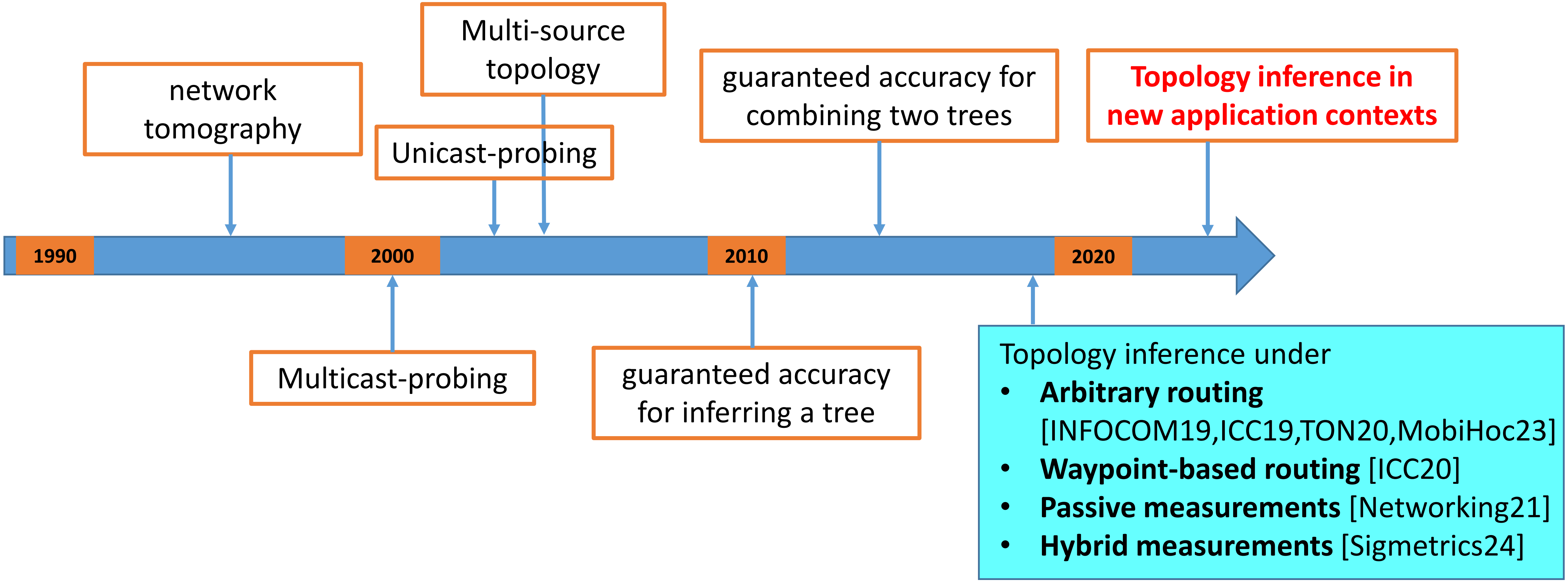
$$-\log\alpha_1 = -\log\left(\frac{\Pr\{X_{p_1} = 1\}\Pr\{X_{p_2} = 1\}}{\Pr\{X_{p_1} = X_{p_2} = 1\}}\right).$$

X_{p_i}: success indicator for path i

No shared link (or sharing a lossless link) ⇒ $-\log\alpha_1 = 0$

Sharing a lossy link ⇒ $-\log\alpha_1 > 0$

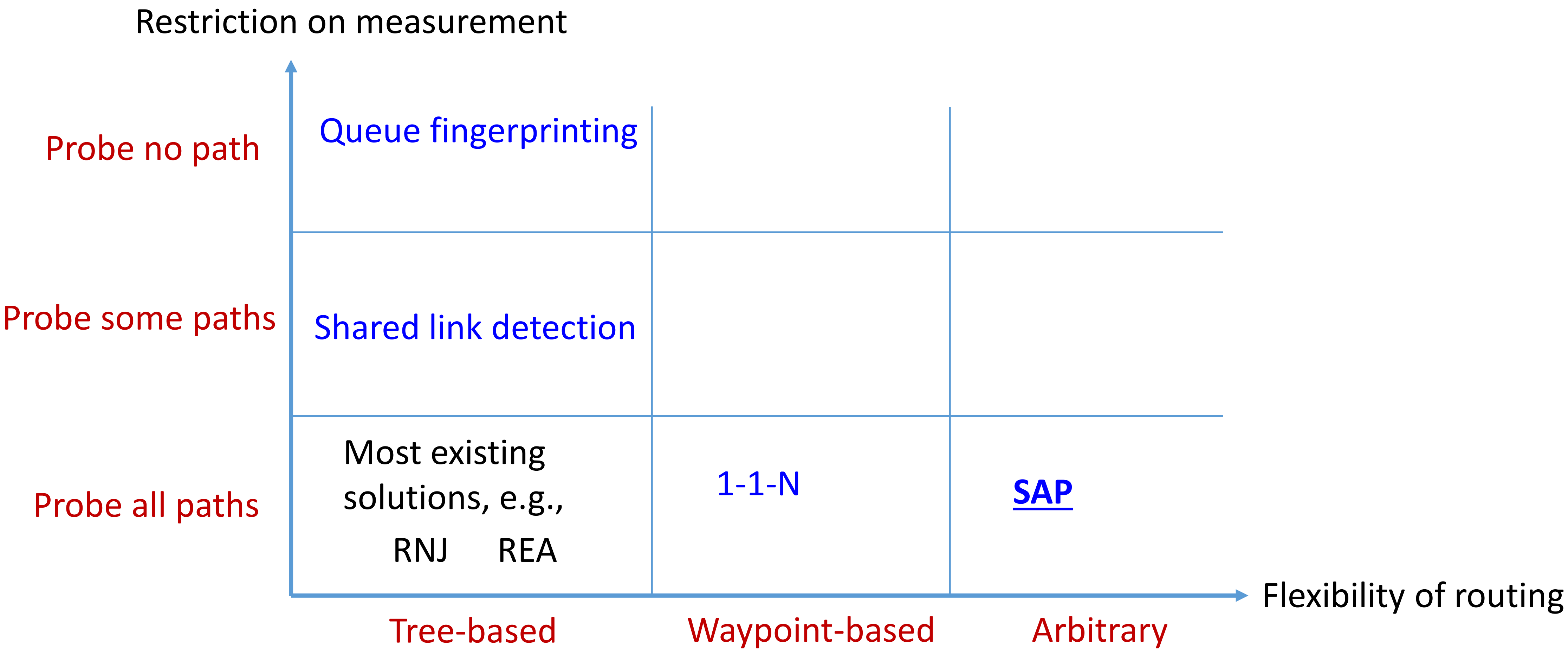
History: Where we are


network tomography
Multi-source topology
Unicast-probing
guaranteed accuracy for combining two trees
Topology inference in new application contexts
1990
2000
2010
2020
Multicast-probing
guaranteed accuracy for inferring a tree

Topology inference under

- **Arbitrary routing** [INFOCOM19,ICC19,TON20,MobiHoc23]
- **Waypoint-based routing** [ICC20]
- **Passive measurements** [Networking21]
- **Hybrid measurements** [Sigmetrics24]

Our approach: Revisiting topology inference problems in new application contexts



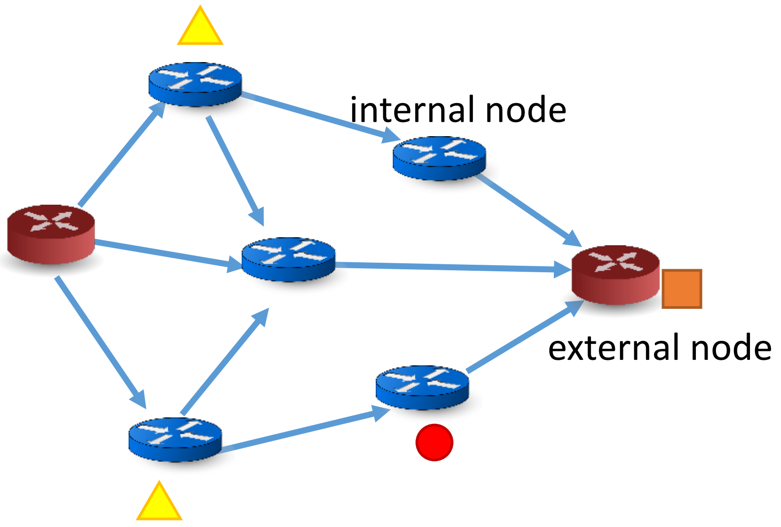
Restriction on measurement

	Tree-based	Waypoint-based	Arbitrary
Probe no path	Queue fingerprinting		
Probe some paths	Shared link detection		
Probe all paths	Most existing solutions, e.g., RNJ REA	1-1-N	**SAP**

Flexibility of routing

Scenario: Probe all paths, arbitrary routing

- **Motivation**: Inferring the structure and state of *SDN-NFV network*
  - general topology
  - waypoint traversal
  - known service chain
- **Observation:**
  - Measured: end-to-end performance measurements (e.g., losses)
  - Inferred: lengths of paths, shared paths, union of paths
    - "length" measured by additive metric
    - E.g., $\theta_e = -\log \alpha_e$ (α_e: success prob. of edge e)
  - Static: source, destination, service chain



NFV network

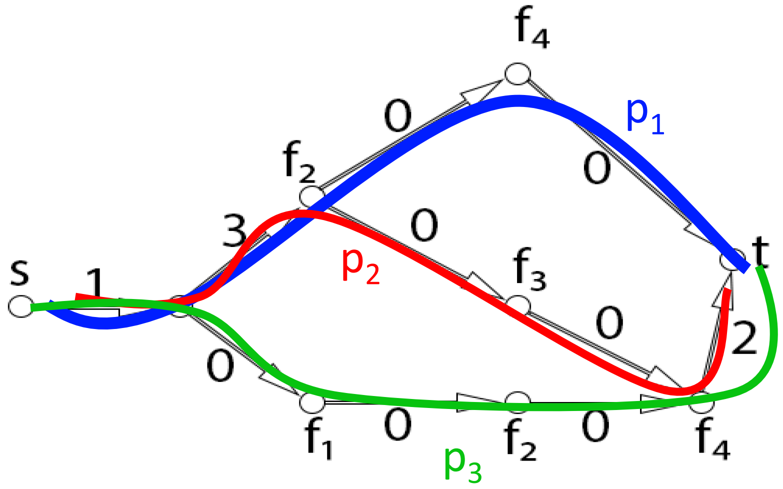
● network function 1 (e.g., IDS)

△ network function 2 (e.g., firewall)

■ network function 3 (e.g., proxy)

Tree-based topology inference is insufficient

- Classic topology inference algorithms all assume tree-based routing
- But trees cannot always reconstruct the observations from a non-tree topology

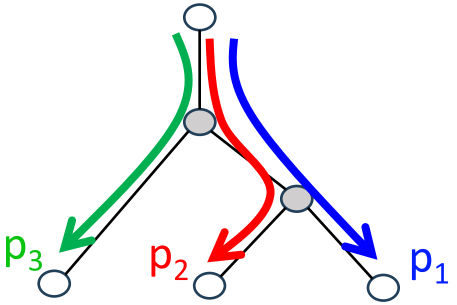


$\text{leng}(p_1)=4$
$\text{leng}(p_2)=6$
$\text{leng}(p_3)=3$

$\text{leng}(p_1 \cap p_2)=4$
$\text{leng}(p_1 \cap p_3)=1$
$\text{leng}(p_2 \cap p_3)=3$

No tree topology reconstructs all these lengths

→ **not even guarantee a feasible solution**



Category weights are identifiable

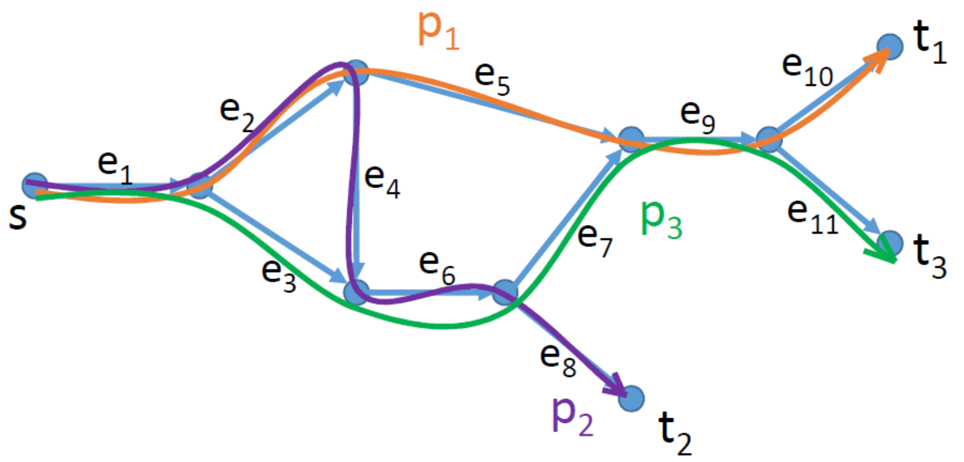
- **Weight Inference Problem**:
  - Partition edges into 2^n-1 *categories*
    - **Category Γ_F**: set of edges *traversed by and only by* paths with indices in F
    - **Category weight w_F**: sum metric of edges in category Γ_F
  - Observe *cast weights*, infer category weights
    - **Cast weight ρ_F** for a multicast on paths in F:

$$\rho_F := -\log(\Pr\{X_F = 1\}) = -\log\left(\prod_{e\in\bigcup_{i\in F} p_i} \alpha_e\right) = \sum_{e\in\bigcup_{i\in F} p_i} \theta_e$$

- Relationship between cast weights and category weights

Topology-agnostic

$$\boxed{\rho_F = \sum_{F'\subseteq E: F'\cap F\neq\emptyset} \mathrm{w}_{F'}, \qquad \forall F\subseteq E}$$



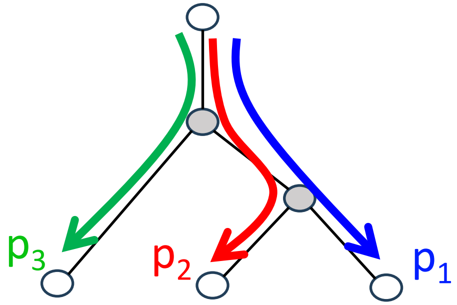
$$\rho_1 = w_1 + w_{1,2} + w_{1,3} + w_{1,2,3}$$
$$\rho_2 = w_2 + w_{1,2} + w_{2,3} + w_{1,2,3}$$
$$\rho_3 = w_3 + w_{1,3} + w_{2,3} + w_{1,2,3}$$
$$\rho_{1,2} = w_1 + w_2 + w_{1,2} + w_{1,3} + w_{2,3} + w_{1,2,3}$$
$$\rho_{1,3} = w_1 + w_3 + w_{1,2} + w_{1,3} + w_{2,3} + w_{1,2,3}$$
$$\rho_{2,3} = w_2 + w_3 + w_{1,2} + w_{1,3} + w_{2,3} + w_{1,2,3}$$
$$\rho_{1,2,3} = w_1 + w_2 + w_3 + w_{1,2} + w_{1,3} + w_{2,3} + w_{1,2,3}$$

Theorem: Category weights are uniquely determined by cast weights.

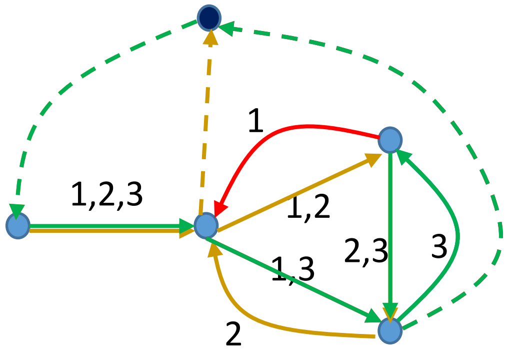
Category weights help, but are not enough

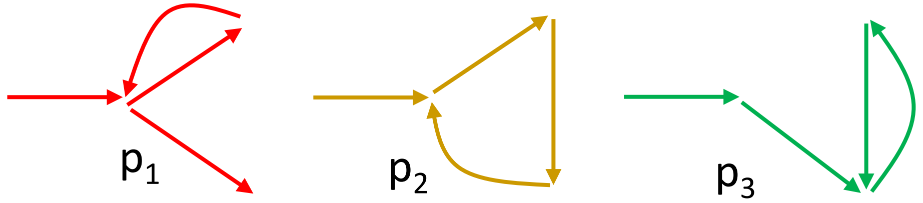
- Under mild assumption, category $\Gamma_F \neq \emptyset \leftrightarrow w_F \neq 0$
- For trees, knowing non-empty categories → knowing (logical) topology

$$\begin{aligned} &\Gamma_{1,2,3} \neq \emptyset \\ &\Gamma_{1,2} \neq \emptyset \\ &\Gamma_{1,}\Gamma_2, \Gamma_3 \neq \emptyset \end{aligned} \quad \Longrightarrow$$



- But not so for arbitrary topology
  - E.g., can always embed the non-empty categories in a clique-like topology



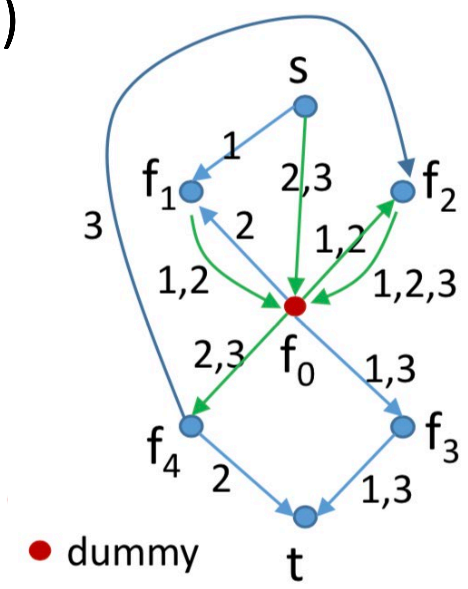


Idea: Combining categories with service chain

- **String Augmentation Problem (SAP):**
  - view each service chain as a string s_i, $f_{i,1}$, $f_{i,2}$,...,t_i
  - insert dummy letters f_0^1, f_0^2,... s.t. for every positive-weight category A, ∃a pair of letters appearing *only* in string i (i∈A)

p'_1: s f_1 f_2 f_3 t
p'_2: s f_2 f_1 f_4 t
p'_3: s f_4 f_2 f_3 t
$\mathcal{A}_+$: {1},{2},{3}, {1,2},{1,3},{2,3}, {1,2,3}

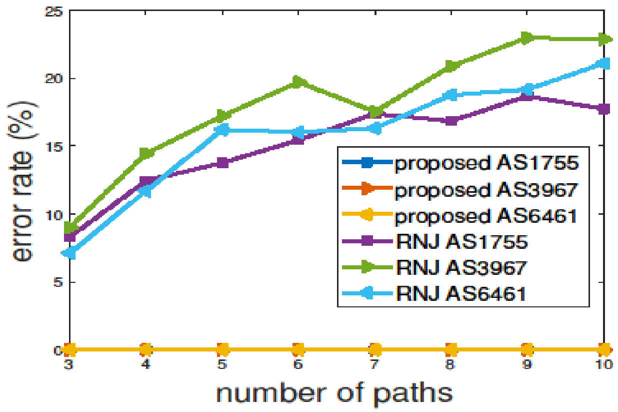
p_1: s f_1 f_0 f_2 f_0 f_3 t
p_2: s f_0 f_2 f_0 f_1 f_0 f_4 t
p_3: s f_0 f_4 f_2 f_0 f_3 t



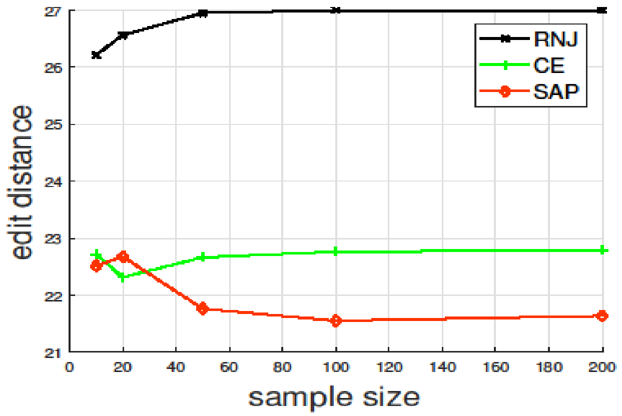
  - Minimize #nodes/#links (can be formulated as an ILP)

Evaluation: VNF topology inference

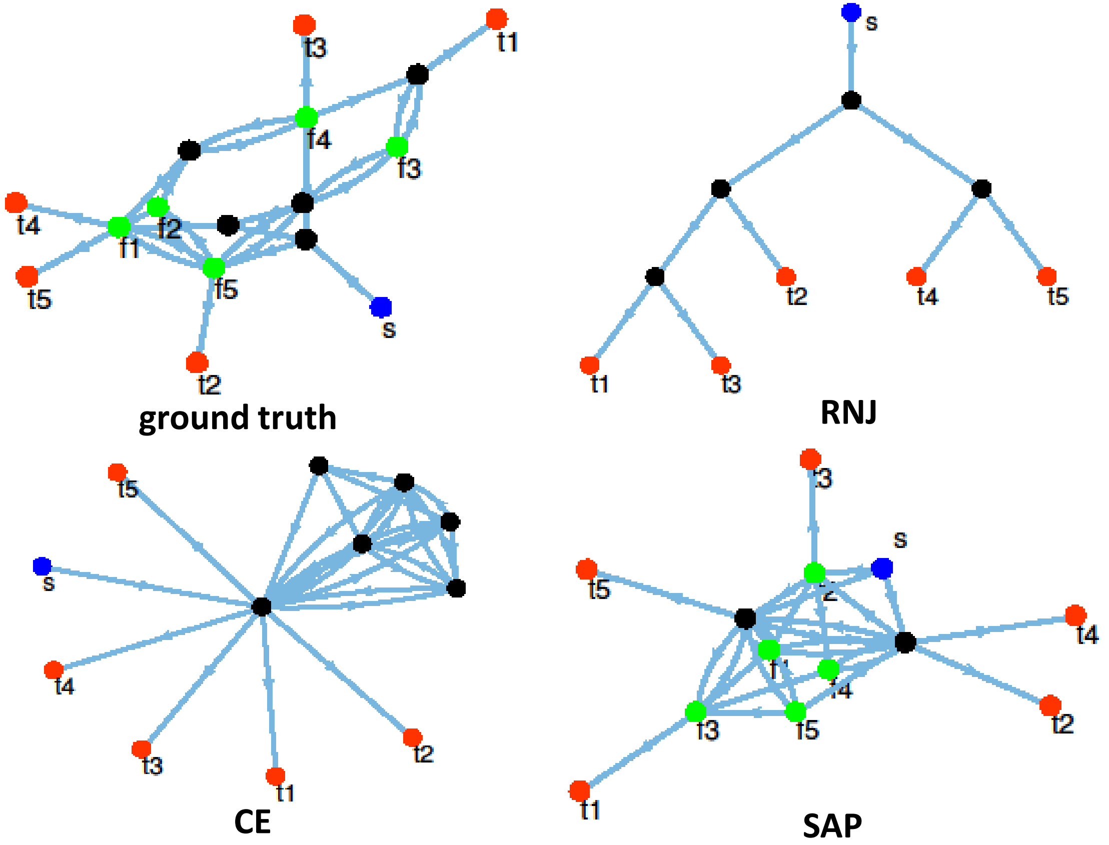
- Based on VNF overlays randomly generated on Rocketfuel AS topologies



(a) reconstruction error



(b) convergence



ground truth

RNJ

CE

SAP

(c) inferred topologies

Topology inference from the perspective of upper-layer application

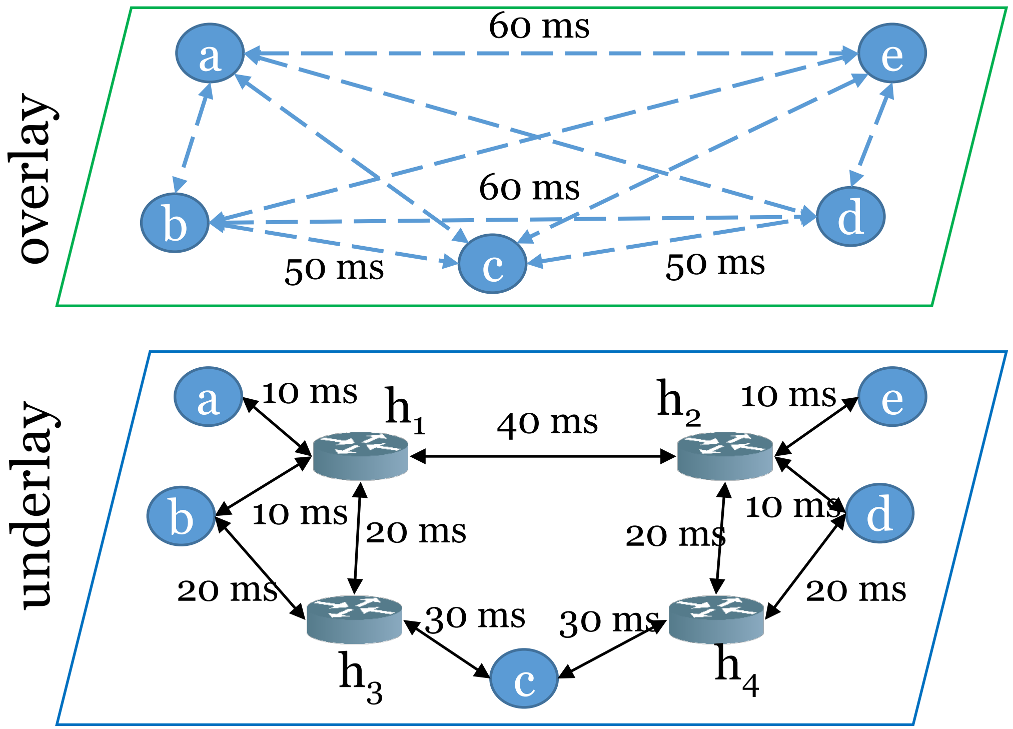
Restriction on measurement

	Tree-based	Waypoint-based	Arbitrary	
Probe no path	Queue fingerprinting			
Probe some paths	Shared link detection			
Probe all paths	Most existing solutions, e.g., RNJ REA	1-1-N	**SAP**	⇨ **Overlay routing**

Flexibility of routing

Overlay Network

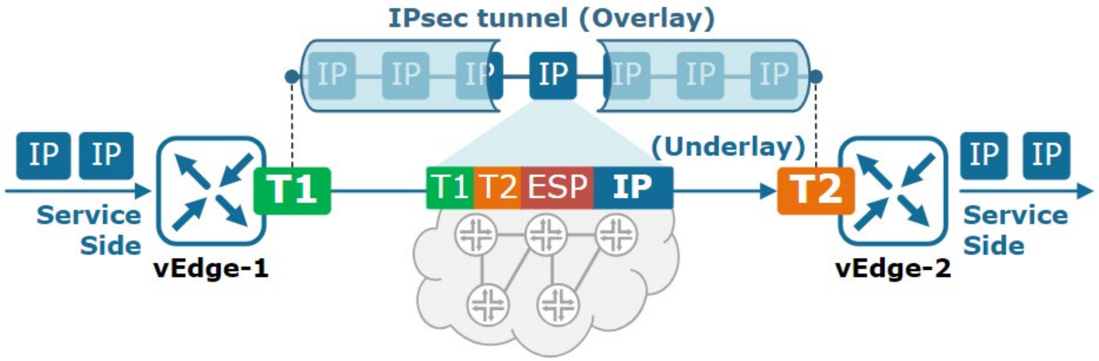
- A logical network running on top of an underlying communication infrastructure (underlay network)
  - Enhance best-effort IP-based underlay network
    - Caching, traffic engineering (service-chaining, multicast), fast failover, network slicing, …
  - Focus: **overlay-based routing**



- Example: SD-WAN
  - Software-Defined Wide-Area Networks

Managed SD-WAN Solutions

AT&T SD-WAN solutions can improve your network's agility and provide centralized control and improve total cost of ownership.

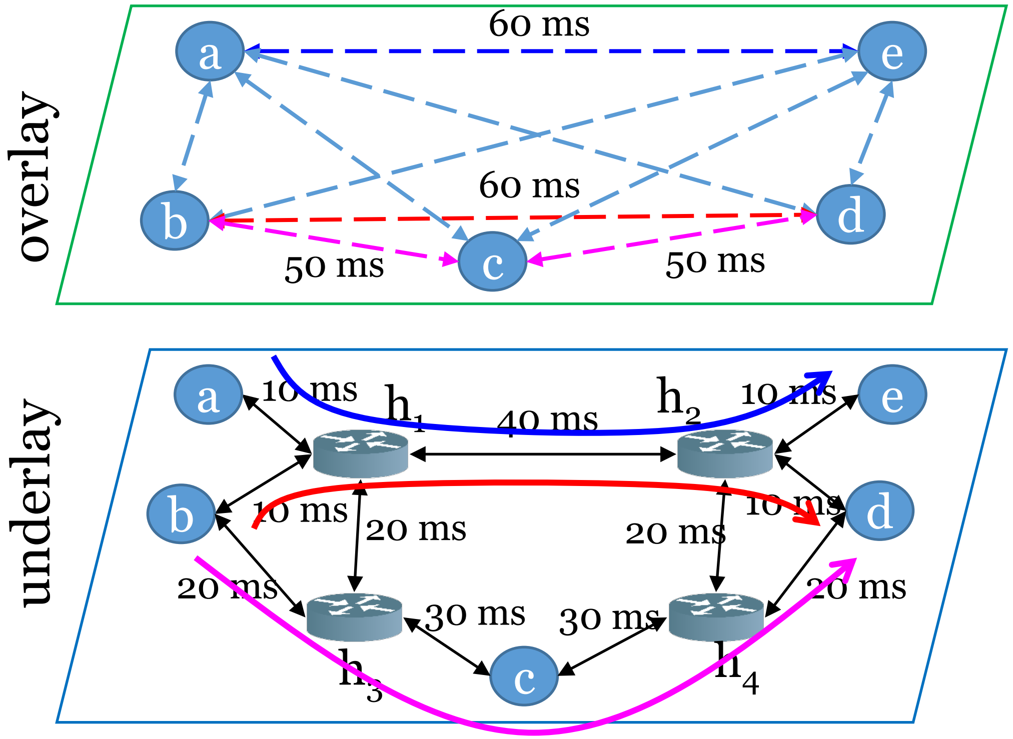


Cisco SD-WAN overlay fabric

Routing in Overlay Network is Challenging

- Challenges
  - Seemingly independent tunnels share underlay links
    - Congestion
  - Uncooperative underlay
    - No direct underlay topology information

Q: Do we need the full topology for overlay routing?
A: No!



- Flow: a->e and b->d
- Direct tunnel: both traverse $h_1 \rightarrow h_2$
- Congestion-free overlay routing:
  - a->e
  - b->c->d

Overlay Routing Problem

$$\min_{\boldsymbol{x}} \sum_{all\_tunnels} tunnel\_cost \sum_{all\_demands} demand \cdot x_{tunnel}^{demand}$$

$$s.t. \quad x_{tunnel}^{demand} \in \{0,1\}$$

flow conservation constraints

Depend on routing & link capacities in underlay

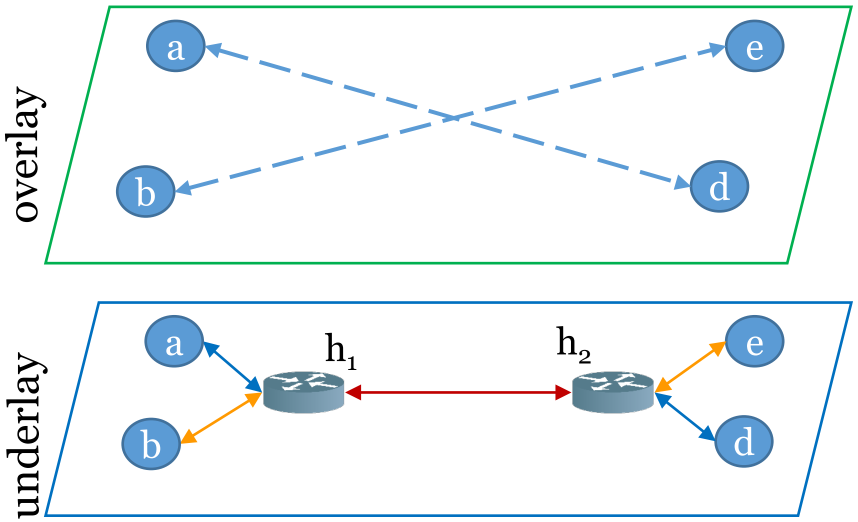
$$\boxed{\sum_{tunnels\_traverse\_link} \sum_{demands} f_{tunnel}^{demand} \leq link\_capacity, \forall links}$$

Q: What is the **minimum information** for **imposing capacity constraints** for an **uncooperative underlay**?

Recall: Underlay Link Categorization

- (Underlay) link category
  - $\Gamma_F(E)$: A **category of links traversed by *F* out of *E*** ($F \subseteq E$) is the set of underlay links traversed **by and only by** the tunnels in F out of all the tunnels in E
    - i.e., $\Gamma_F(E) \coloneqq \underbrace{\boxed{\left(\bigcap_{(i,j)\in F} \underline{p}_{i,j}\right)}}_{\text{Links shared by } F} / \underbrace{\boxed{\left(\bigcup_{(i,j)\in E\setminus \mathrm{F}} \underline{p}_{i,j}\right)}}_{\text{All links traversed by } E\setminus F}$
  - Category weight: $w_F(E) \coloneqq \sum_{\underline{e}\in\Gamma_F(E)} \theta_{\underline{e}}$

Observation: Knowledge of **link categories suffices for congestion-free overlay routing**



Example: $E = \{(a,d),(b,e)\}$

- $F_1 = \{(a,d),(b,e)\}$
  - $\Gamma_{F_1}(E) = \{(h_1,h_2)\}$
- $F_2 = \{(a,d)\}$
  - $\Gamma_{F_2}(E) = \{(a,h_1),(h_2,d)\}$
- $F_3 = \{(b,e)\}$
  - $\Gamma_{F_3}(E) = \{(b,h_1),(h_2,e)\}$

Category-based Capacity Constraints

Links in the same category receive the same traffic load from the overlay

Full topology information – which tunnels traverse each link

Per-link constraints:

$$\sum_{tunnels\_traverse\_link}\sum_{demands} f_{tunnel}^{h} \le link\_capacity$$

Full capacity information – what is the capacity of each link

Partial topology information – which tunnels exclusively share links, i.e., $\Gamma_F(E) \neq \emptyset$

Per-category constraints:

$$\sum_{tunnels\_in\_category}\sum_{demands} f_{tunnel}^{h} \le category\_capacity$$

Partial capacity information – what is the min link capacity in each category

Challenge of Category Inference

Measurements in overlay → ρ_F

$$\rho_F := \sum_{\underline{e} \in \bigcup_{(i,j)\in F} \underline{p}_{i,j}} \theta_{\underline{e}}$$

Candidate category weight w_F

$$w_F(E) := \sum_{\underline{e} \in \Gamma_F(E)} \theta_{\underline{e}}$$

$$\boxed{\rho_F = \sum_{F' \subseteq E : F' \cap F \neq \emptyset} \mathrm{w}_{F'}(E)\,, \forall F \subseteq E}$$

- **Full rank** linear system
- $w_F(E) > 0 \Longrightarrow \Gamma_F(E) \neq \emptyset$

Q: Is problem solved?
A: Unfortunately, **no**

- **Exponential complexity**! #variables = $2^{|E|} = 2^{O(|V|^2)}$
- Example: $|V| = 10$, number of candidate categories: 2^{90}

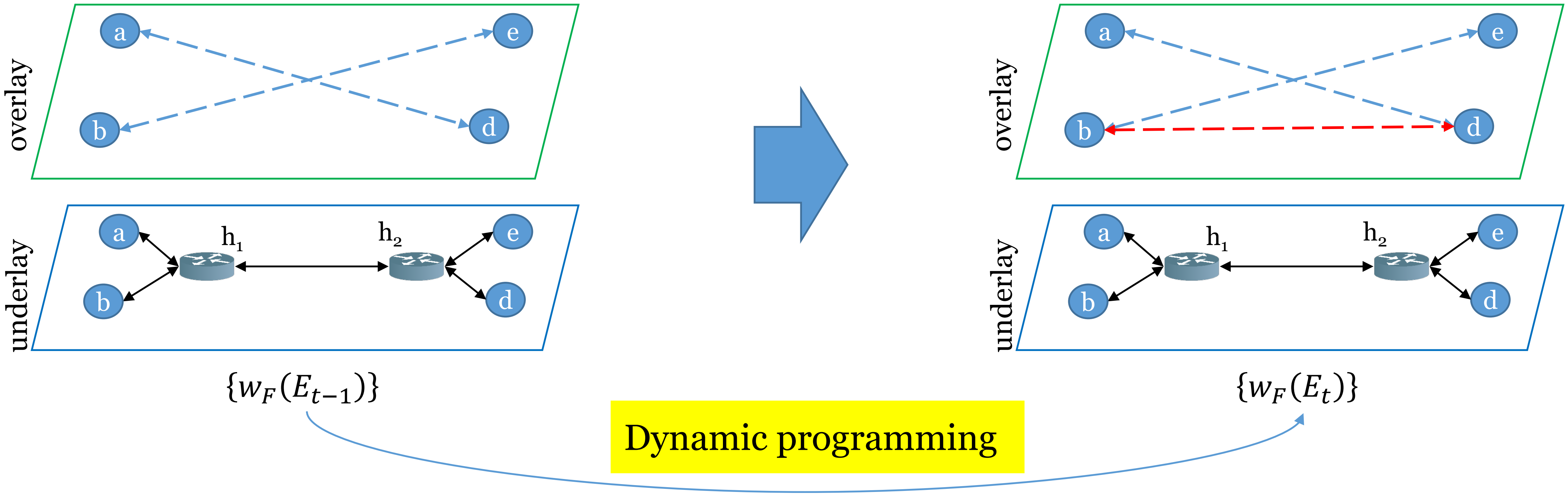
Taming the Complexity in Category Inference

Idea: Given $\{w_F(E_{t-1})\}$ and $E_t \leftarrow E_{t-1} \cup \{e_t\}$, augment it into $\{w_F(E_t)\}$

→ Dynamic programming

$E_{t-1} = \{(a,d),(b,e)\}$

$E_t = \{(a,d),(b,e),(b,d)\}$



$\{w_F(E_{t-1})\}$

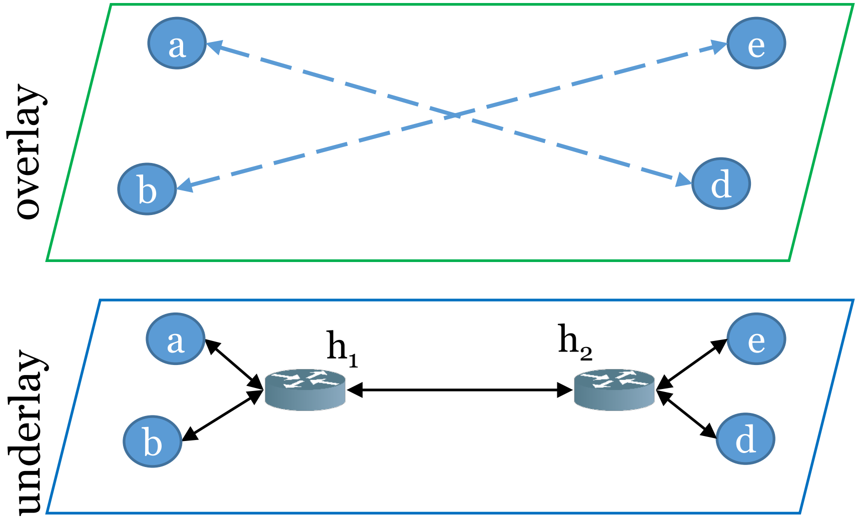
$\{w_F(E_t)\}$

Dynamic programming

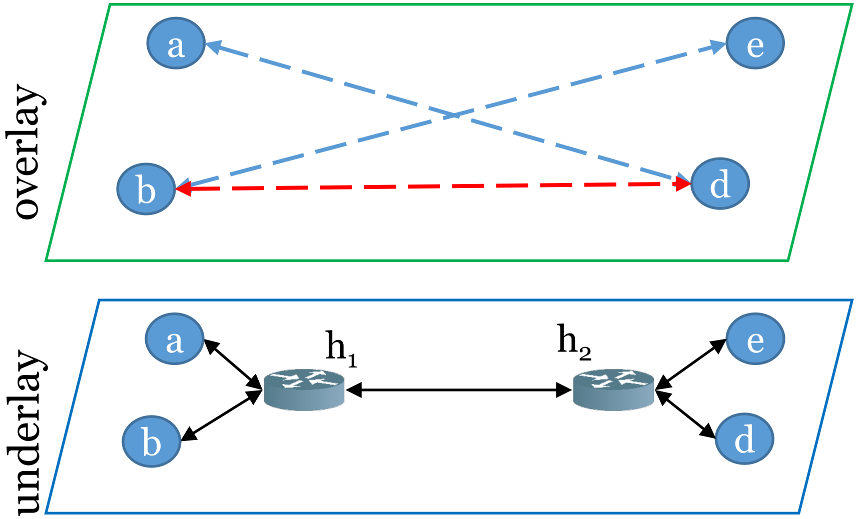
Idea for Dynamic Programming

- Category weights are decomposed gradually
  - For any $E' \subset E$ and $e \in E \setminus E'$, $w_F(E') = w_{F\cup\{e\}}(E' \cup \{e\}) + w_F(E' \cup \{e\})$

$E_{t-1} = \{(a,d),(b,e)\}$



$E_t = \{(a,d),(b,e),(b,d)\}$



$\{w_F(E_{t-1})\}$

$\{(a,d),(b,e)\}$ → $\{(a,d),(b,e),(b,d)\},\{(a,d),(b,e)\}$

$\{(a,d)\}$ → $\{(a,d),(b,d)\},\{(a,d)\}$

$\{(b,e)\}$ → $\{(b,e),(b,d)\},\{(b,e)\}$

$\{(b,d)\}$

$\{w_F(E_t)\}$ candidates

$|supp(\mathbf{w}(E_{t-1}))| \le |\underline{E}|$

(#non-empty categories ≤ #underlay links)

#variables = $2|supp(\mathbf{w}(E_{t-1}))| + 1$

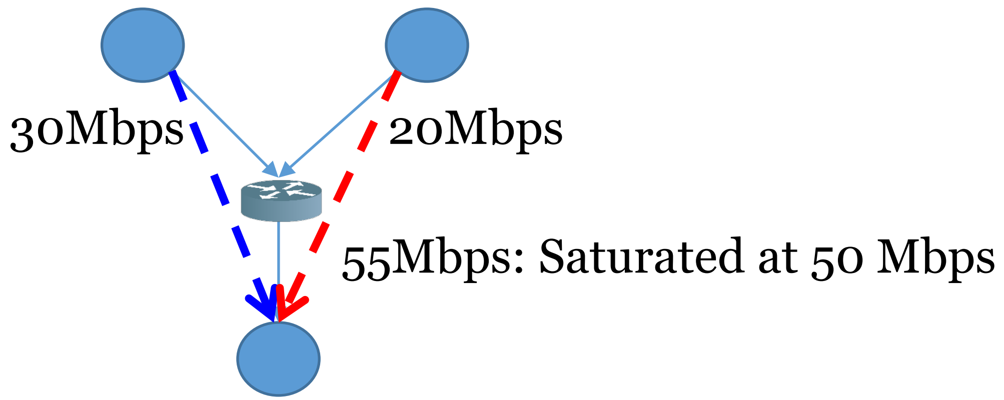
Algorithm for Category Inference

- Dynamic programming with the update rule:
  - $E_t \leftarrow E_{t-1} \cup \{e\}$
  - $w_{\{e\}}(E_t) \leftarrow \rho_{E_t} - \rho_{E_{t-1}}$
  - For $F \in supp\big(\boldsymbol{w}(E_{t-1})\big)$ in an increasing order of $|F|$:
    - $w_{F\cup\{e\}}(E_t) \leftarrow \rho_{(E_{t-1}\setminus F)\cup\{e\}} - \rho_{E_{t-1}\setminus F} - w_{\{e\}}(E_t) - \sum_{F'\subset F: F\in supp(\boldsymbol{w}(E_{t-1}))} w_{F'\cup\{e\}}(E_t)$
    - $w_F(E_t) \leftarrow w_F(E_t) - w_{F\cup\{e\}}(E_t)$
  - #variables $= 2\left|supp\big(\boldsymbol{w}(E_{t-1})\big)\right| + 1 = O\big(|\underline{E}|\big)$

- In each iteration, solve a **linear system** whose size is **linear in the underlay network size**.
- In total $|E|$ iterations, **linear in the overlay network size**
- The first **polynomial-time** algorithm for category inference

Effective Category Capacity Inference

- The minimum capacity of the links in a category may not be measurable



- Effective Category Capacity: maximum flow through the tunnels associated with the category
  - $\tilde{C}_F \coloneqq \max_{(f_e)_{e \in E}} \sum_{e \in F} f_e$ (f_e: flow assigned to tunnel e)

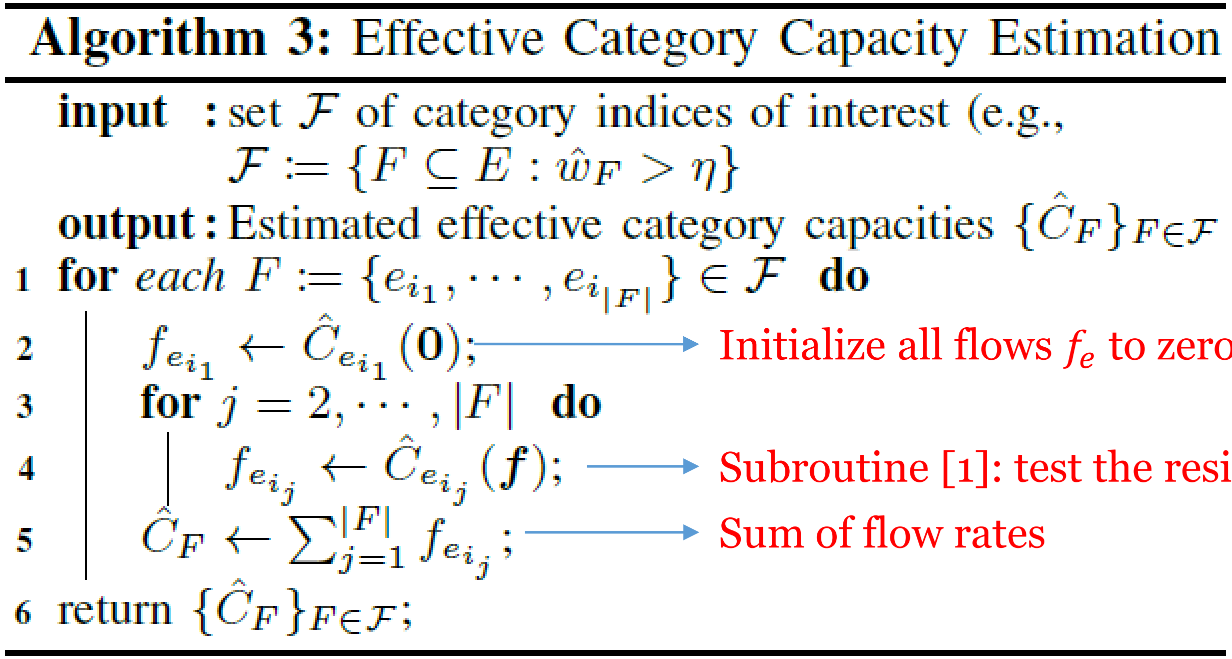
    s.t. $\sum_{e' \in F'} f_{e'} \leq \boxed{C_{F'}}, \forall F' \subseteq E, \Gamma_{F'} \neq \emptyset$

    $f_e \geq 0, \forall e \in E$ UNKNOWN

Effective Category Capacity Estimation

- Algorithm:

[1] Jain M, Dovrolis C. “End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput,” IEEE/ACM TNET, 2003.



Algorithm 3: Effective Category Capacity Estimation

```
input  : set $\mathcal{F}$ of category indices of interest (e.g.,
         $\mathcal{F} := \{F \subseteq E : \hat{w}_F > \eta\}$
output : Estimated effective category capacities $\{\hat{C}_F\}_{F \in \mathcal{F}}$
1 for each $F := \{e_{i_1}, \cdots, e_{i_{|F|}}\} \in \mathcal{F}$ do
2     $f_{e_{i_1}} \leftarrow \hat{C}_{e_{i_1}}(\mathbf{0})$;
3     for $j = 2, \cdots, |F|$ do
4         $f_{e_{i_j}} \leftarrow \hat{C}_{e_{i_j}}(\boldsymbol{f})$;
5     $\hat{C}_F \leftarrow \sum_{j=1}^{|F|} f_{e_{i_j}}$;
6 return $\{\hat{C}_F\}_{F \in \mathcal{F}}$;
```

Initialize all flows f_e to zero

Subroutine [1]: test the residual capacity of a tunnel given flow assignment

Sum of flow rates

- Performance guarantee
  - If Line~4 is accurate, then Algorithm 3 achieves ${}^{1}/_{q_F}$ approximation
    - $q_F := \max_{e \in F} |\{F' \subseteq E : e \in F', \Gamma_{F'} \neq \emptyset, |F' \cap \mathrm{F}| > 1\}|$
    - maximum number of nonempty categories a tunnel in F traverses that are shared by at least another tunnel in F

Resulting Overlay Routing Problem

$$\min_{\boldsymbol{x}} \sum_{all\_tunnels} tunnel\_cost \sum_{all\_demands} demand \cdot x_{tunnel}^{demand}$$

$$s.t. \quad x_{tunnel}^{demand} \in \{0,1\}$$

flow conservation constraints

$$\sum_{\boxed{tunnels\_in\_category}} \sum_{demands} f_{tunnel}^{h} \le \boxed{category\_capacity}$$

Partial topology information

– which tunnels exclusively share links

Partial capacity information

– what is the effective category capacity

NS3-Based Simulation

- Topologies from Internet Topology Zoo

	AttMpls	AboveNet	GTS-CE	BellCanada
$\lvert \underline{V} \rvert$	25	23	149	48
$\lvert \underline{E} \rvert$	114	62	386	130
$C_{\underline{e}}$ (Gbps)	1	1	1	1
Link delays (us)	[206,4973]	[100, 13800]	[5,1081]	[78, 6160]

- Background traffic
  - ON-OFF process for each link independently
    - Duration follows Pareto distribution
    - Utilization: [10%,40%]
- Probing
  - Number of overlay nodes: 10
  - 50-byte packets for probing; 1000-byte packets for routings
  - Measurements: end-to-end delays
- Routing cost: link (propagation) delays

Performance of Inference

Non-Empty Category Detection

	AttMpls	AboveNet	GTS-CE	BellCanada
#empty cat.	$2^{90} - 69$	$2^{90} - 52$	$2^{90} - 59$	$2^{90} - 51$
#nonempty cat.	69	52	59	51
#false alarms	603	542	2159	1695
#misses	20	27	40	30

- **Low false alarm rate** although the absolute number is not small
- **High miss rate:** Inaccurate estimation of ρ_F if (1) $|F|$ is large or (2) tunnels in F have different sources

Effective Category Capacity Estimation

	AttMpls	AboveNet	GTS-CE	BellCanada
ideal subroutine	0.10%	0.13%	0.13%	0.4%
Pathload	1.07%	1.18%	1.15%	1.49%

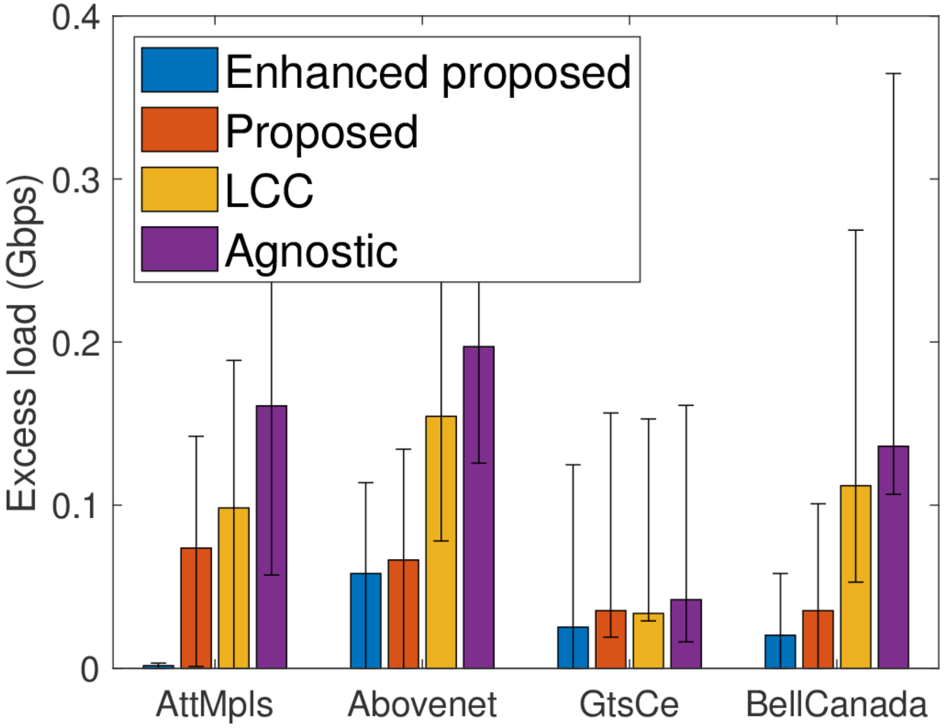
- **Highly accurate capacity estimation:** ***False alarms will not hurt*** in most case, but ***misses may lead to congestions***.

Performance of Overlay Routing

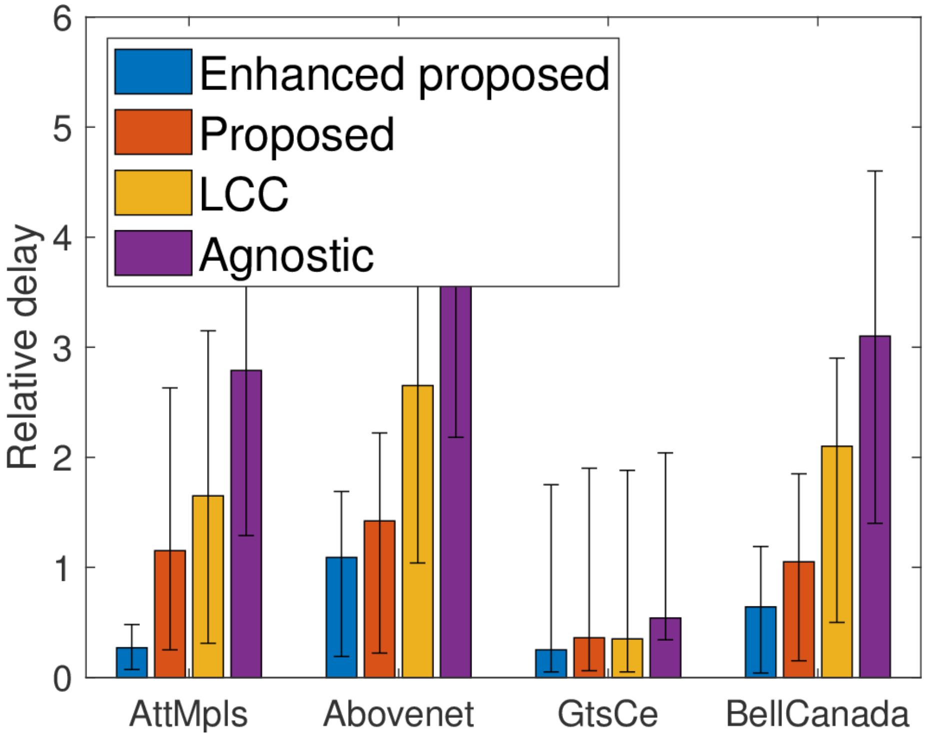
[2] Y. Zhu and B. Li, “Overlay networks with linear capacity constraints,” IEEE TPDS, 2008

- Benchmarks
  - “**Agnostic**”: an underlay-agnostic routing
  - “**LCC**”: the state-of-the-art solution from [2]
  - “**Proposed**”
  - “**Enhanced proposed**”: “Proposed” + “LCC”

Improved overlay routing performance despite notable estimation errors



max excess load

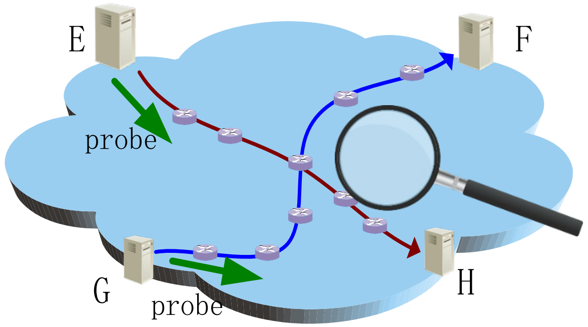


avg relative delay

Concluding Remark

- Topology inference: **Jointly infer network *internal structure & state* from *external observations***
  - What structures are possible, what measurements are allowed

  → A tool for **application-layer network optimization** (e.g., overlay routing)



Network structure & state = ?

Restriction on measurement

	Tree-based	Waypoint-based	Arbitrary
Probe no path	Queue fingerprinting		
Probe some paths	Shared link detection		
Probe all paths	Most existing solutions, e.g., RNJ REA	1-1-N	SAP

Flexibility of routing

CT Scan for Your Network: Topology Inference from End-to-End Measurements

Ting He, tinghe@psu.edu

THANK YOU

Backup slides

Outline

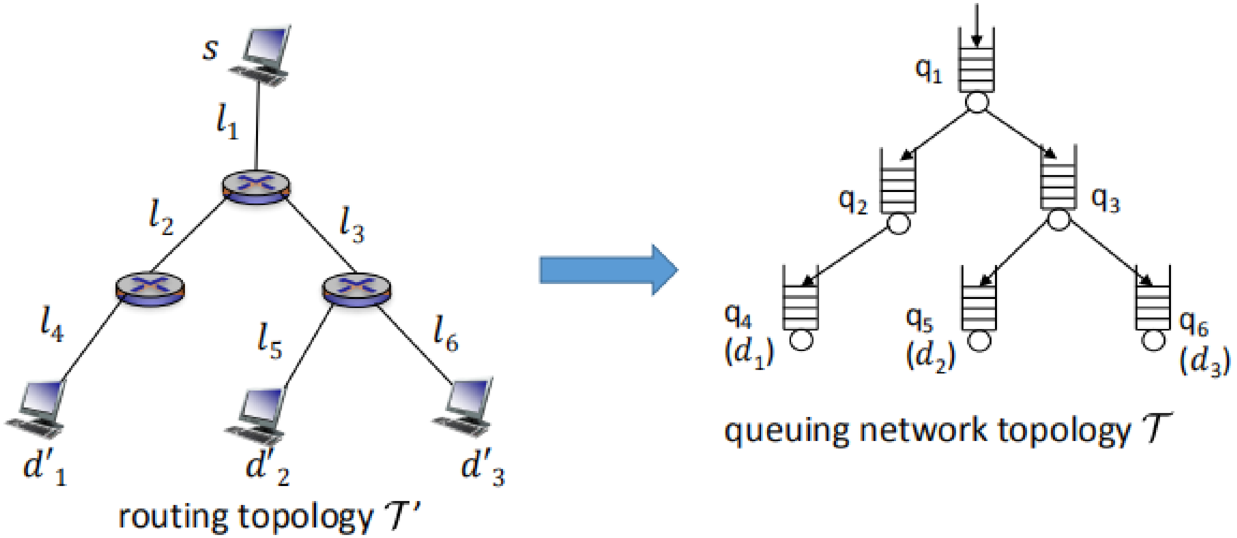
Restriction on measurement

	Tree-based	Waypoint-based	Arbitrary
Probe no path	**Queue fingerprinting**		
Probe some paths	Shared link detection		
Probe all paths	Most existing solutions, e.g., RNJ REA	1-1-N	SAP

Flexibility of routing

Scenario: Passive monitoring only

- A network of independent M/M/1 queues



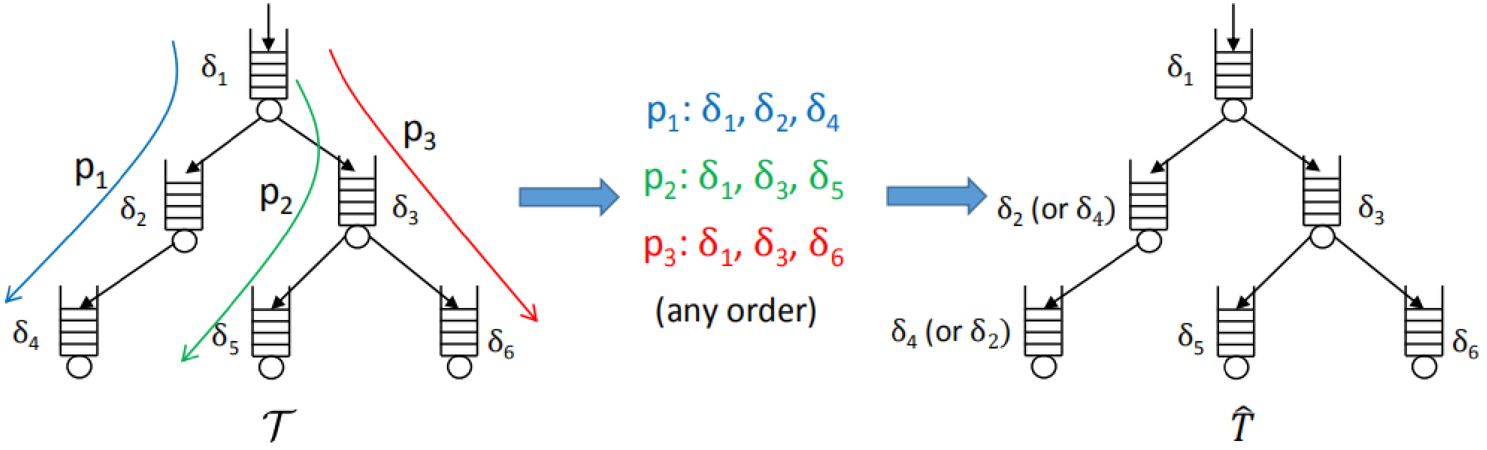
routing topology $\mathcal{T}'$

queuing network topology $\mathcal{T}$

- **Goal**: Address two key limitations of existing solutions
  - Active probing → **passive monitoring**
  - Logical topology → **physical topology**

Why it is feasible

- Queue parameter: $\delta_i = \mu_i - \lambda_i$ (residual capacity)
- Sojourn time: exponential r.v. with PDF $\delta_i e^{-\delta_i t_i}$
- End-to-end delay: hypoexponential r.v. with parameters $\boldsymbol{\delta} \coloneqq (\delta_i)_{i=1}^{K}$
- Idea: **Queue fingerprinting**



p_1: $\delta_1, \delta_2, \delta_4$

p_2: $\delta_1, \delta_3, \delta_5$

p_3: $\delta_1, \delta_3, \delta_6$

(any order)

(a)

(b) Estimated parameters

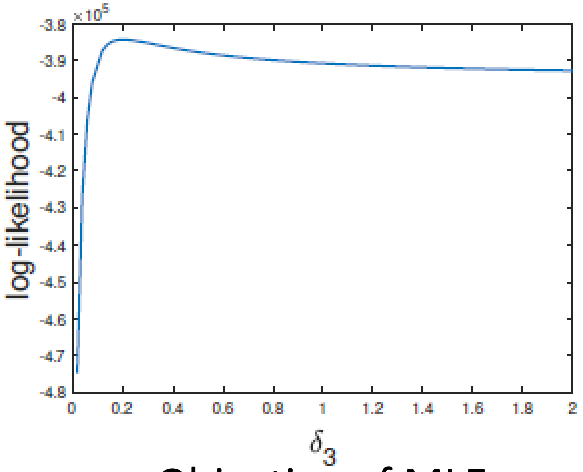
(c) Inferred topology

Parameter estimation for tandem of M/M/1 queues: Estimator

- Idea 1: **MLE**

$$\boxed{\hat{\boldsymbol{\delta}} = argmax_{\boldsymbol{\delta}} \sum_{h=1}^{n} \log g(x_h; \boldsymbol{\delta})}$$

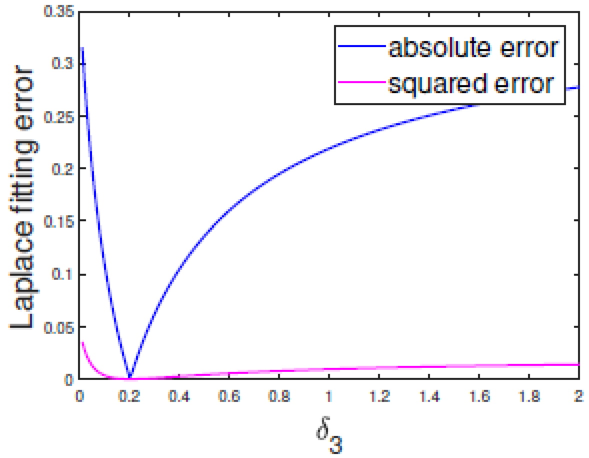
  - PDF: $$g(x;\ \boldsymbol{\delta}) = \sum_{i=1}^{K} \delta_i e^{-x\delta_i} \left(\prod_{j=1,j\neq i}^{K} \frac{\delta_j}{\delta_j - \delta_i} \right)$$



Objective of MLE

- Idea 2: **Fitting Laplace transform**
  - Laplace transform: $$L(s;\ \boldsymbol{\delta}) := \prod_{i=1}^{K} \frac{\delta_i}{\delta_i + s}, \qquad s > -\min_{i=1,\ldots,K} \delta_i.$$
  - Empirical Laplace transform: $$\hat{L}(s;\ \boldsymbol{x}) := \frac{1}{n} \sum_{h=1}^{n} e^{-s x_h}$$

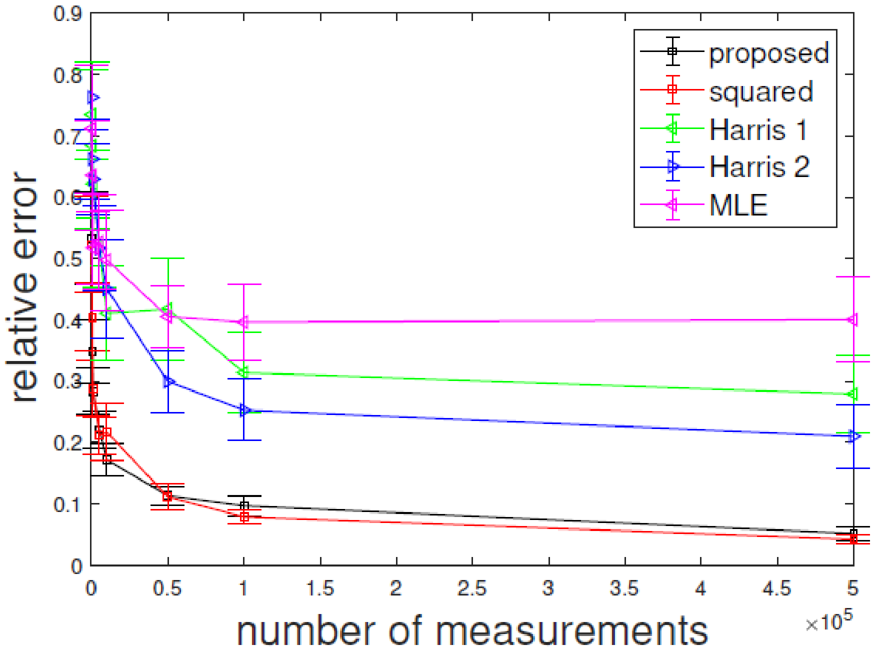
→ $$\boxed{\begin{aligned} \min \quad & \sum_{s \in S} \left| L(s;\ \boldsymbol{\delta}) - \hat{L}(s;\ \boldsymbol{x}) \right| \\ \text{s.t. } & 0 < \delta_1 \leq \cdots \leq \delta_K, \end{aligned}}$$



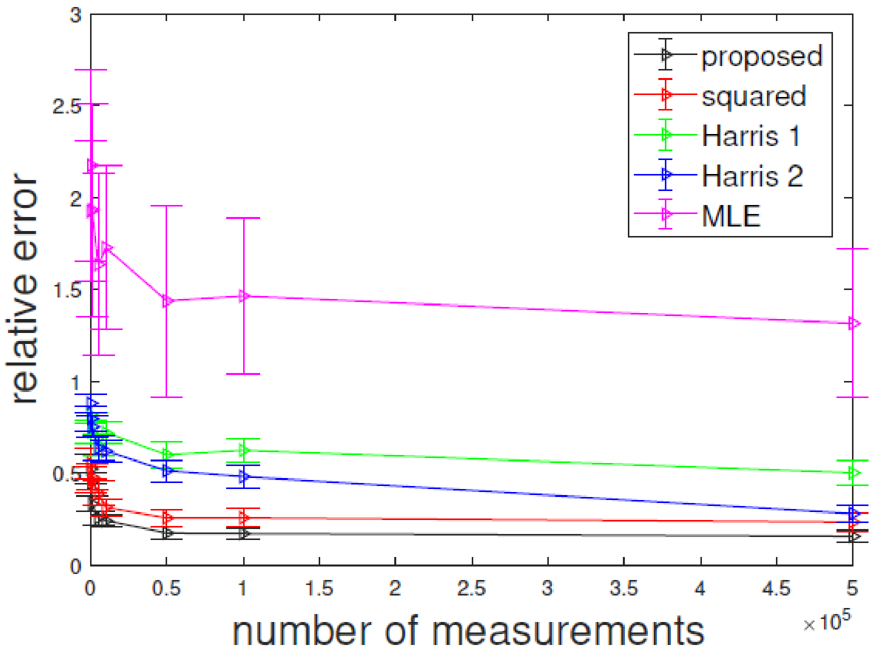
Objective of Laplace fitting

Parameter estimation for tandem of M/M/1 queues: Performance

- **Theorem.** As $n \rightarrow \infty$, Laplace fitting has a unique optimal solution that equals the ground truth $\boldsymbol{\delta}$ if $|S| > K$.



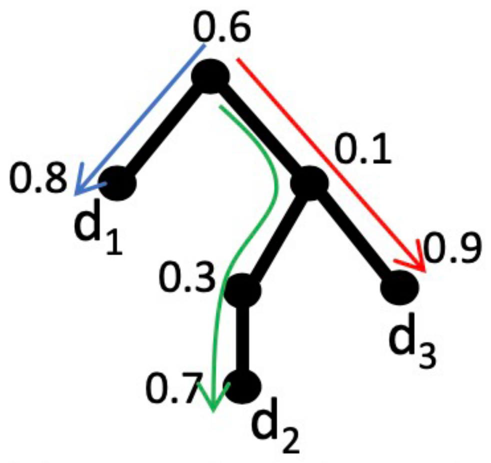
K = 3



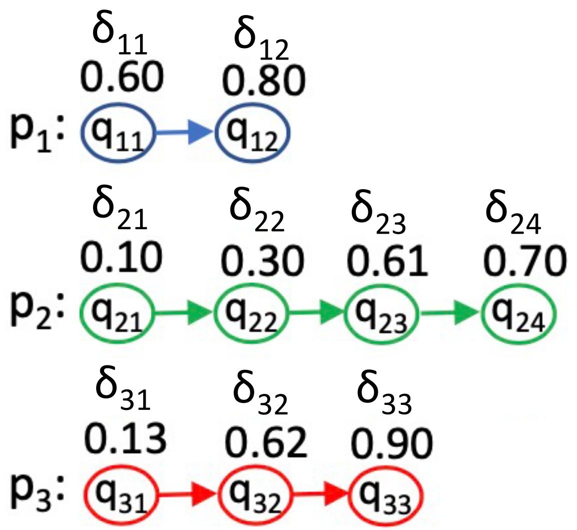
K = 4

Queueing topology inference: idea

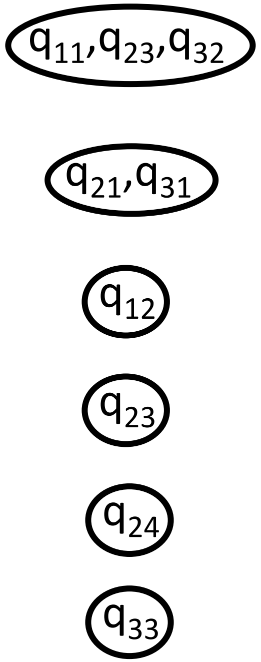
- Ideal case:



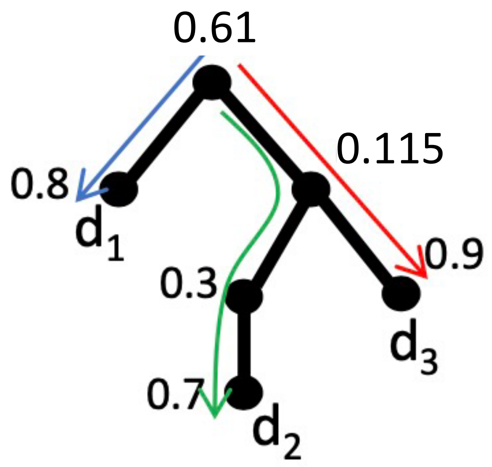
Ground truth topology



Estimated parameters



Parameters associated with the same queue



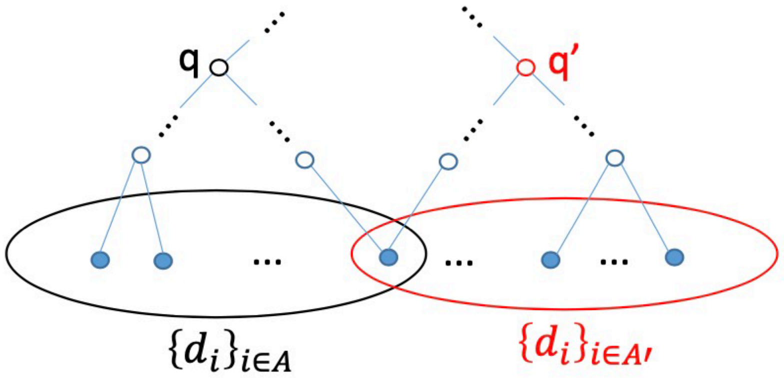
Inferred topology

Queueing topology inference: challenges

- Parameter estimation is not perfect
  - An upper bound Δ, such that

$$D_{\{q_{i_1 j_1},\ldots,q_{i_k j_k}\}} := \max\{\delta_{i_1 j_1},\ldots,\delta_{i_k j_k}\} - \min\{\delta_{i_1 j_1},\ldots,\delta_{i_k j_k}\} \leq \Delta$$

- Topology is not arbitrary
  - Partially overlapping categories cannot coexist
- Exponential complexity if brute-forcing
  - $O(K^N)$ ways to merge queues



Queueing topology inference: solution

- A *greedy* algorithm with *progressively constructed search space* to infer estimated parameters associated with the same queue
  - $O(K^4 N^5)$ time complexity, $O(K^2 N^3)$ space complexity
  - Correct if estimated parameters are sufficiently accurate

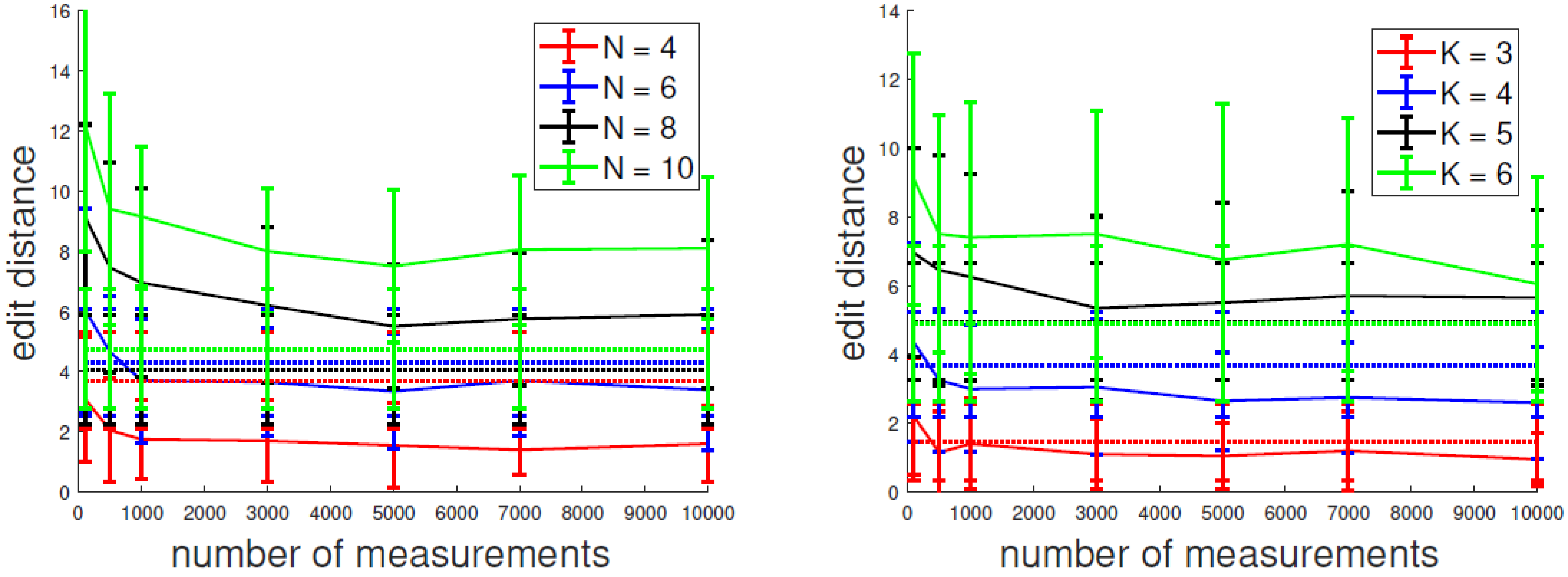
- **Theorem.** All parameters for the same queue are correctly identified if

$$\left|\delta_{ij} - \delta_{ij}^*\right| \leq \frac{\Delta}{2} < \frac{\Delta^*}{4} \quad \text{(where } \Delta^* := \min_{e \neq e\prime} |\delta_e^* - \delta_{e\prime}^*|\text{)}$$

→ Under this condition**, the inferred topology will be identical to the ground truth**, up to a permutation of queues on the same branch.

Performance evaluation

- Routing trees generated from AS6461 of Abovenet



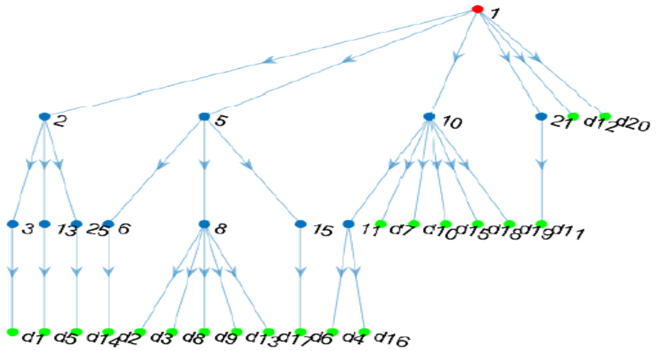
(a) vary N $(K = 4)$

(b) vary K $(N = 5)$

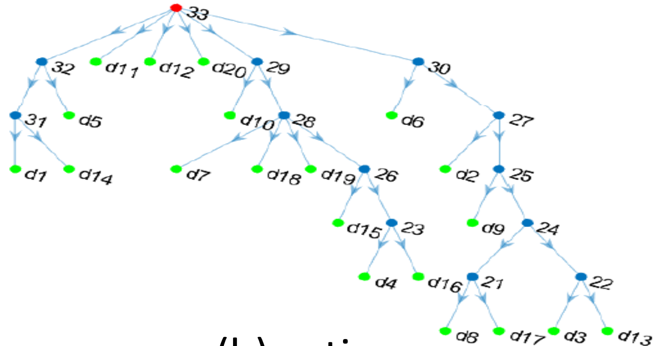
solid line: edit distance for inferred topology; dotted line: edit distance for multicast tree

How to improve the scalability

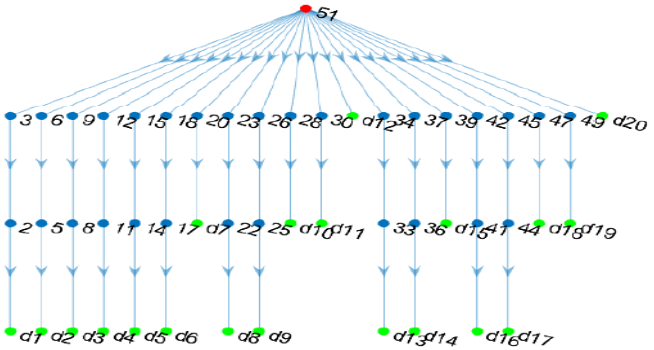
- Idea: Combining passive & active measurements
  - Passive measurements → queue fingerprints
  - Active measurements → shared path length



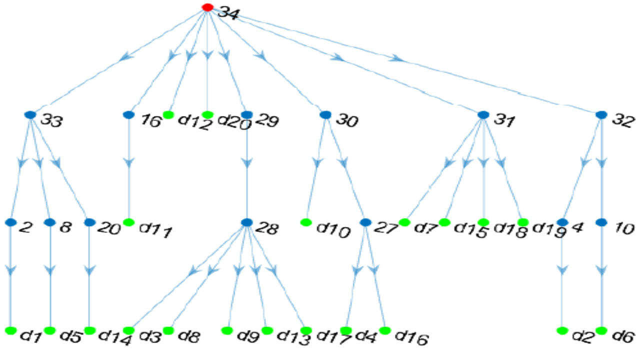
a) ground truth



(b) active

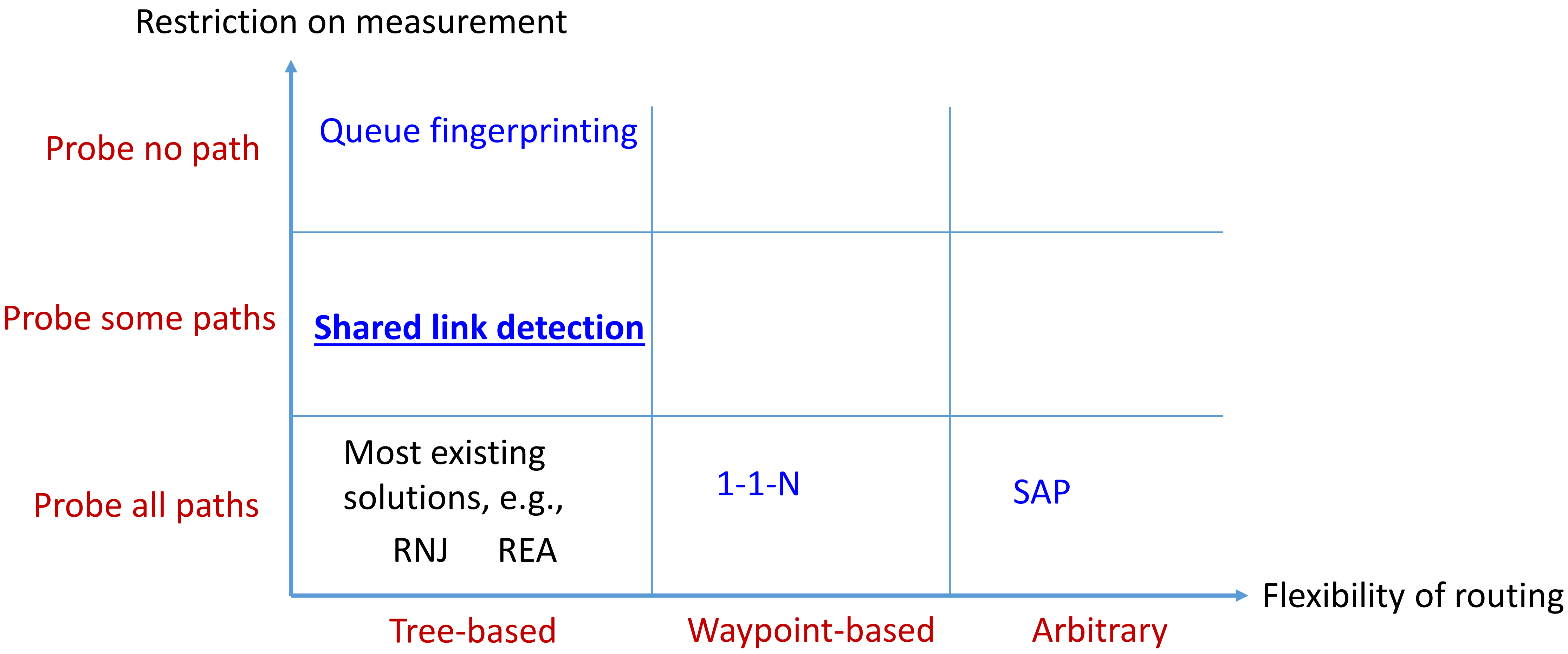


(c) passive



(d) combined

Outline



Restriction on measurement

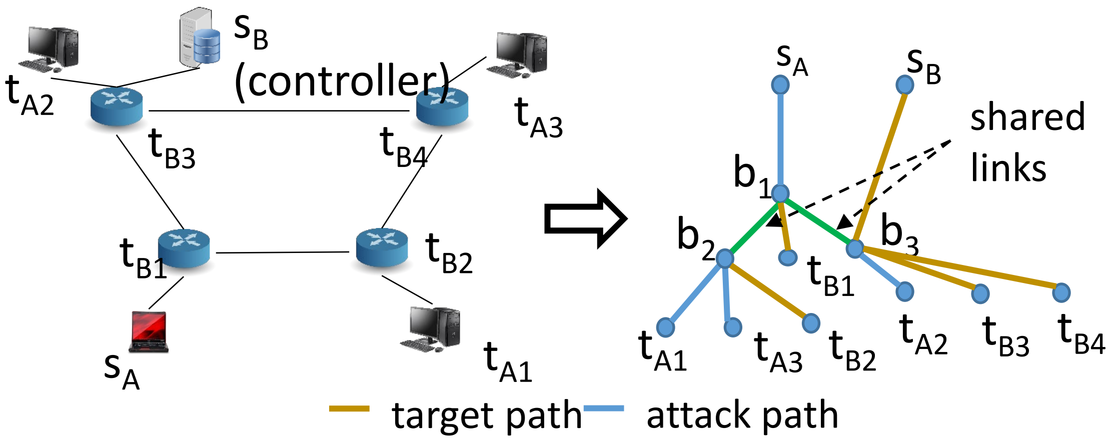
	Tree-based	Waypoint-based	Arbitrary
Probe no path	Queue fingerprinting		
Probe some paths	**Shared link detection**		
Probe all paths	Most existing solutions, e.g., RNJ REA	1-1-N	SAP

Flexibility of routing

Scenario: Cross-path attack

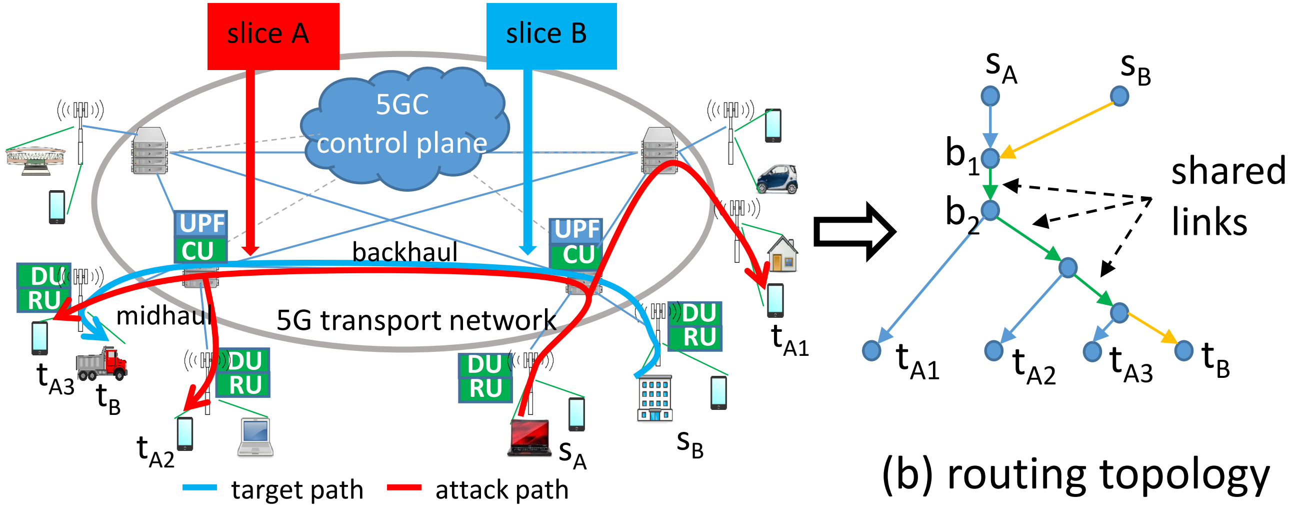
- An attacker in control of a set of *attack paths* wants to launch indirect DoS attack on a set of *target paths* by consuming shared resources

Example 1: Data→Control Plane Attack in SDN



(a) physical network (b) routing topology

Example 2: Cross-slice Attack in 5G



(a) physical network

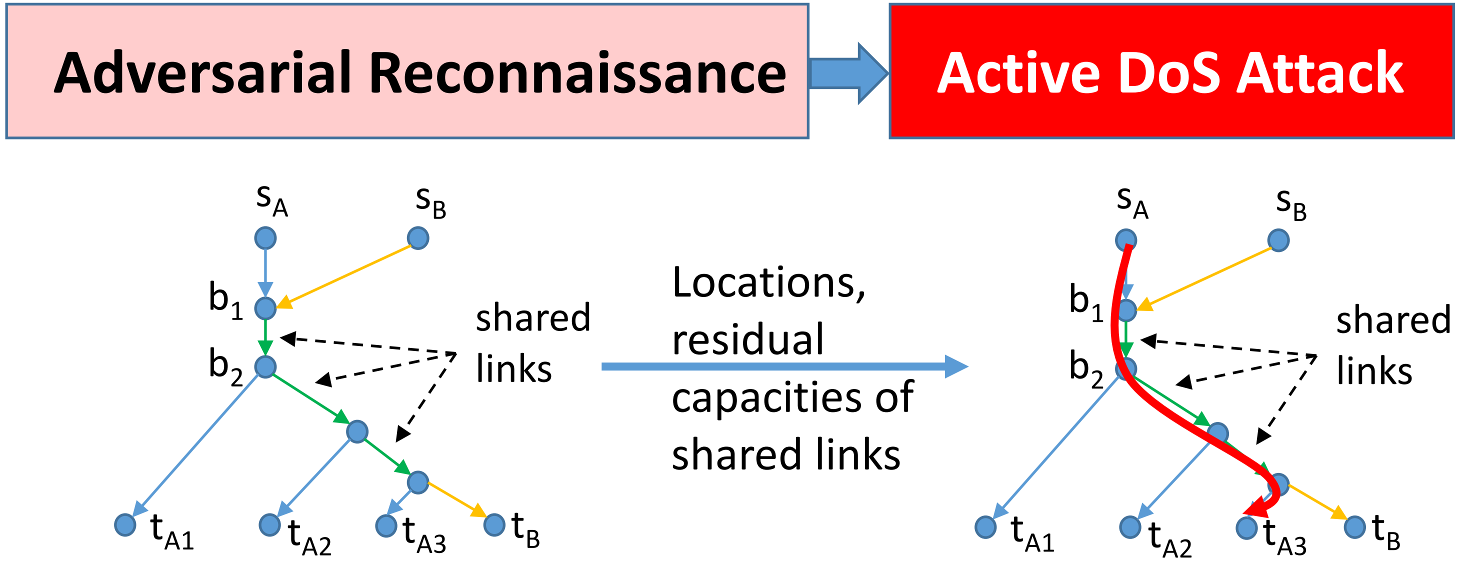
(b) routing topology

Cross-path attack: A high-level description

- Cross-path attack contains a *reconnaissance phase* and an *active attack phase*

Which attack paths share resource with target paths?
What is the capacity of the shared resource?

Which attack paths to use?
How much traffic to send?

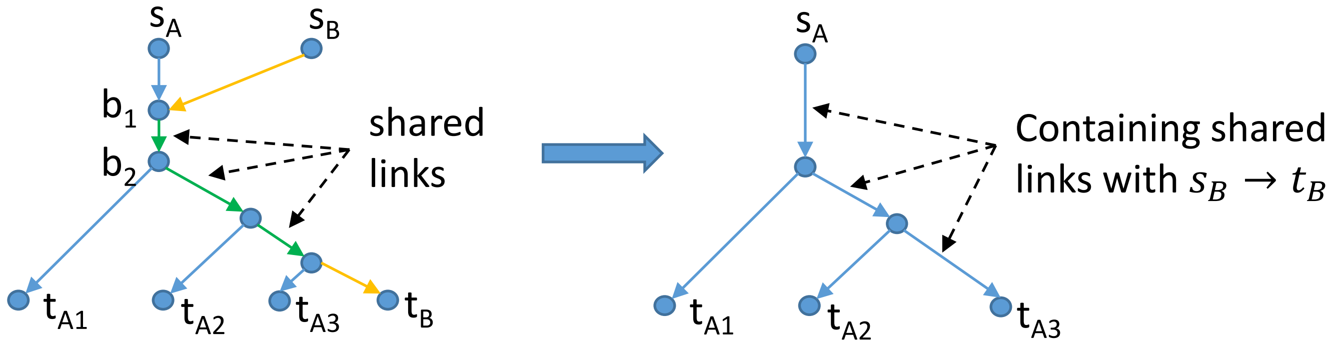


Adversarial Reconnaissance → **Active DoS Attack**

Locations, residual capacities of shared links

Adversarial reconnaissance: A topology inference problem

- **Observation model**: *Active probing* on attack paths, *passive monitoring* on target paths
- **Goal:** Support optimal attack design
  - Knowing the true routing topology formed by all attack/target paths is sufficient, but not necessary
- **Idea:** Use mimicked multicast to infer "attack paths + 1 target path" topologies



Adversarial reconnaissance: Results

- Recursive algorithm to detect shared links
  - **Theorem.** If all shared links have non-zero metrics and **category weights are estimated accurately**, then all **shared links will be correctly detected**.

- Recursive algorithm to estimate parameters of detected shared links
  - Modeled as M/M/1, M/D/1, or G/G/1 queue
  - Estimated by fitting average delay under K different probing rates
  - **Theorem.** If all shared links are correctly detected, and the **average delays on target paths are accurately estimated**, then the **parameters of shared links will be accurately estimated** if (i) $K > 2$ under M/M/1 or M/D/1, and (ii) $K > 4$ under G/G/1

Attack design: Objectives and results

- Objective 1: Delay maximization

$$\max f(\bar{\boldsymbol{\lambda}}) := \sum_{i=1}^{N_B} \beta_i \sum_{e\in\mathcal{T}:W_{ie}>0} d(\xi_{ie}; \sum_{k=1}^{N_A} h_{ek}\bar{\lambda}_k)$$

$$\text{s.t.} \quad \sum_{k=1}^{N_A} \bar{\lambda}_k \leq \lambda,$$

$$\sum_{k=1}^{N_A} h_{ek}\bar{\lambda}_k \leq \tilde{r}_e, \ \forall e \in \mathcal{T},$$

$$\bar{\lambda}_k \geq 0, \ k = 1, \ldots, N_A,$$

- Objective 2: Overload maximization

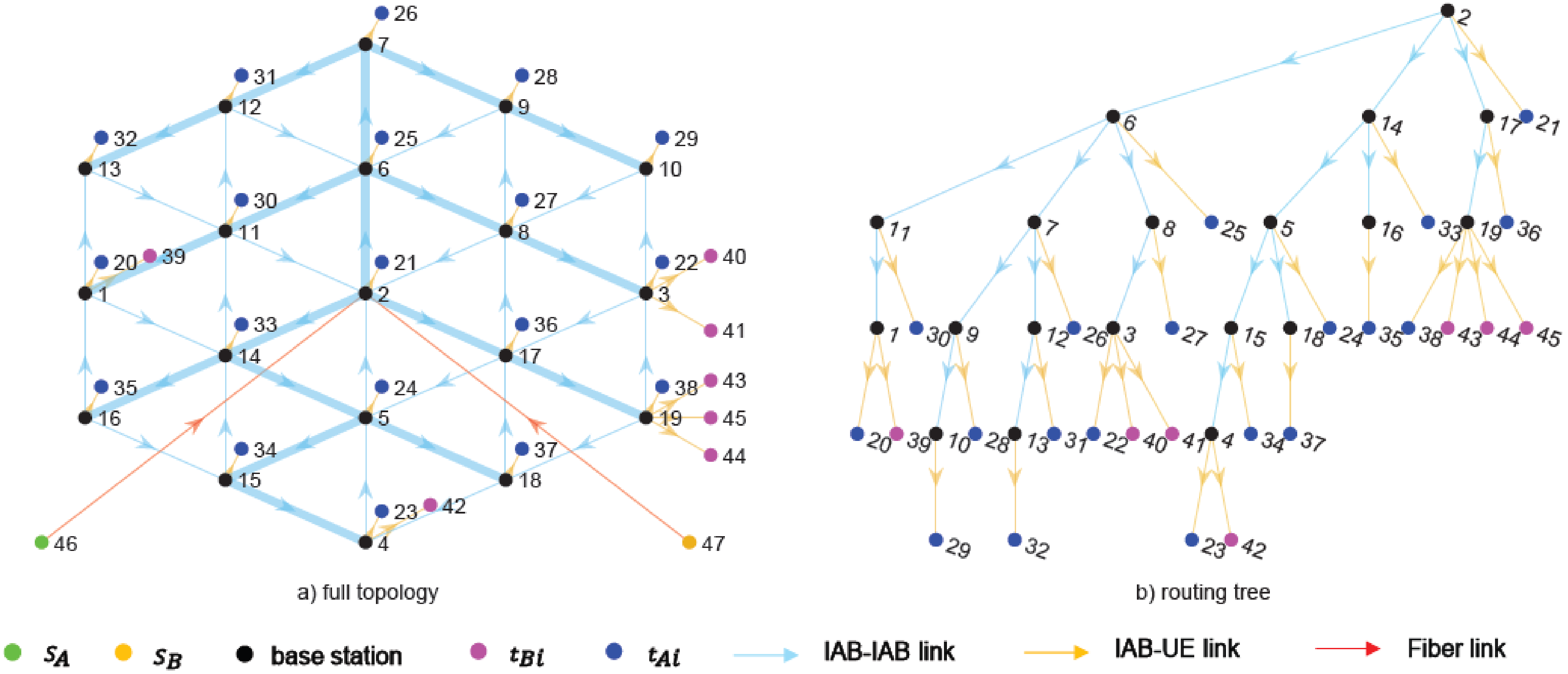
$$\max_{\bar{\boldsymbol{\lambda}}} \max_{e\in\mathcal{T}:\exists W_{ie}>0} \left(\sum_{k=1}^{N_A} h_{ek}\bar{\lambda}_k - \min_{i\in\{1,\ldots,N_B\}:W_{ie}>0} r_{ie} \right)$$

$$\text{s.t.} \ \sum_{k=1}^{N_A} \bar{\lambda}_k \leq \lambda, \quad \sum_{k=1}^{N_A} h_{ek}\bar{\lambda}_k \leq \tilde{r}_e, \ \forall e \in \mathcal{T}, \quad \bar{\lambda}_k \geq 0, \ k = 1, \ldots, N_A,$$

Both **maximizing convex function under linear constraints**
→ Optimum at a vertex
→ If attack rate $\lambda \leq \min_{e\in T} \tilde{r}_e$, optimal to send all attack traffic on one attack path

Performance evaluation: NS3 + 5G Lena

- Scenario: 5G IAB (Integrated Access and Backhaul) network

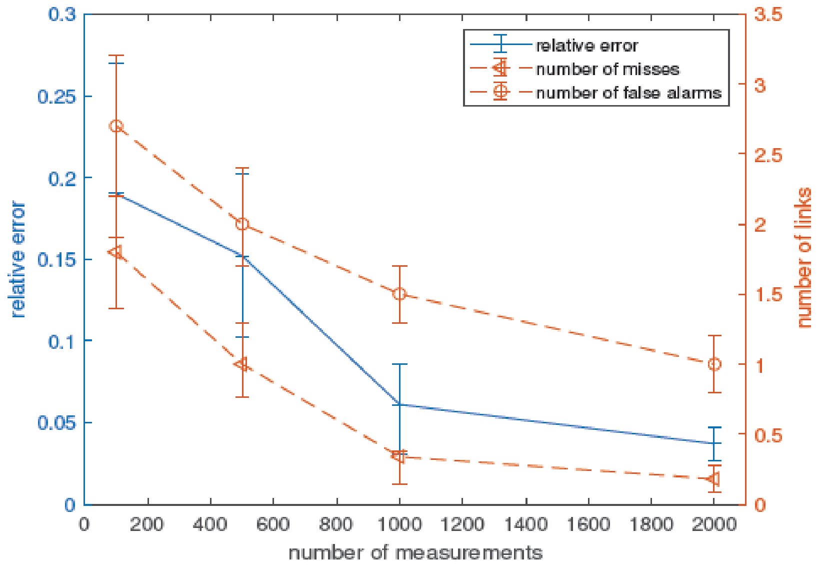


a) full topology

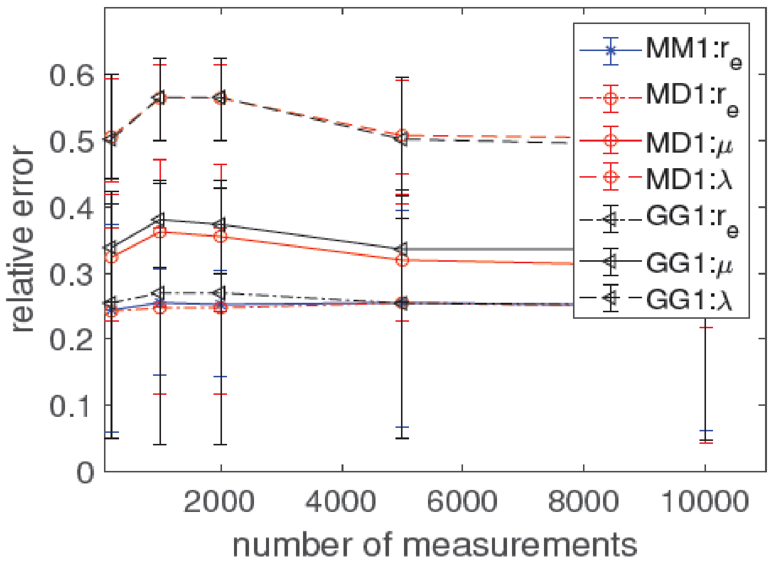
b) routing tree

- ON-OFF traffic, discrete packet sizes

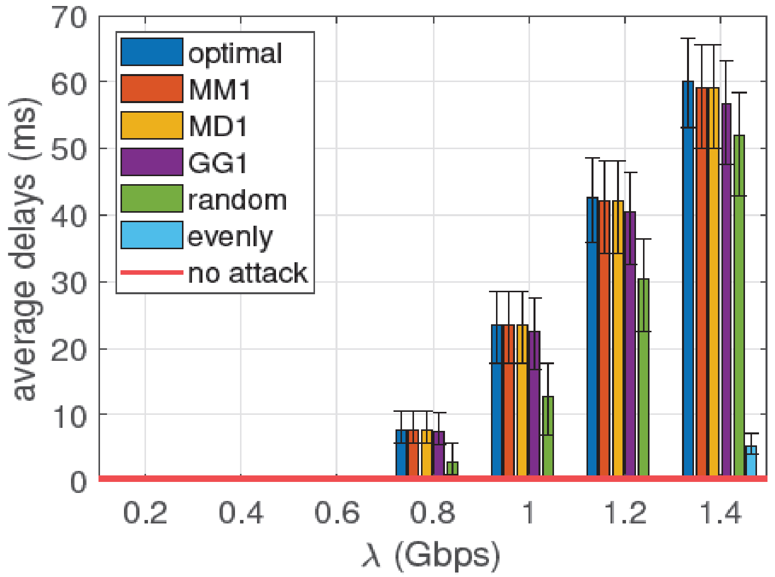
Performance evaluation: Results



(a)



(b)



(c)

a) Can detect most of the shared links

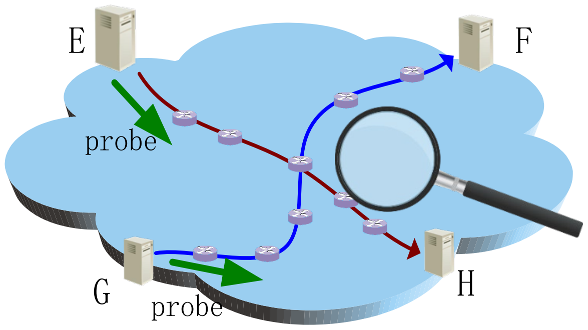
b) Notable error in estimated parameters

c) Near-optimal performance in attack design

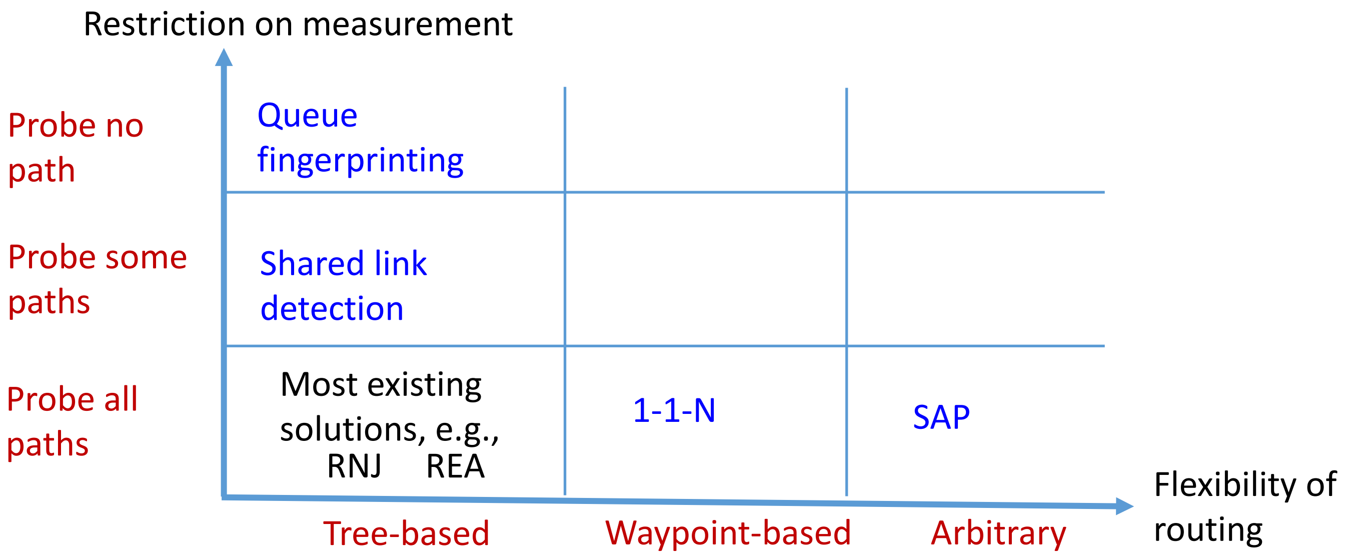
Concluding Remark

- Topology inference: Jointly infer network *internal structure* from *external observations*
  - what “internal structure” to infer, what structures are possible, what measurements are allowed

  → A double-sided sword (overlay management vs. adversarial reconnaissance)



Network structure & state = ?



Restriction on measurement

	Tree-based	Waypoint-based	Arbitrary
Probe no path	Queue fingerprinting		
Probe some paths	Shared link detection		
Probe all paths	Most existing solutions, e.g., RNJ REA	1-1-N	SAP

Flexibility of routing

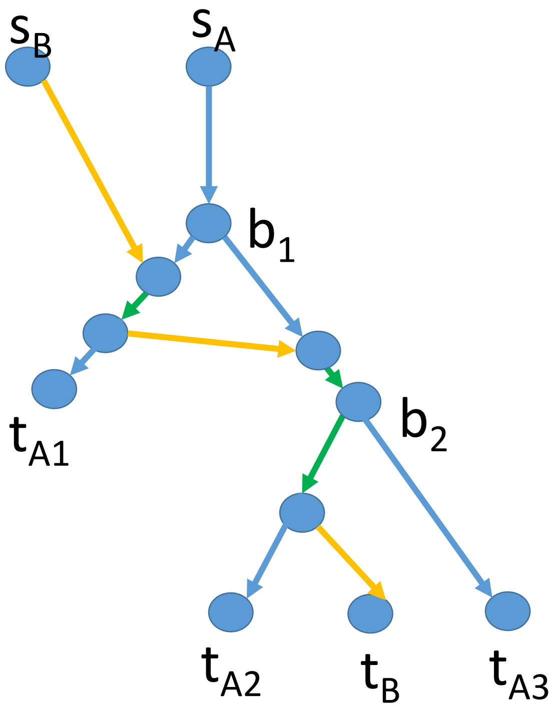
CT Scan for Network: Topology Inference from End-to-End Measurements

Ting He, tinghe@psu.edu

THANK YOU

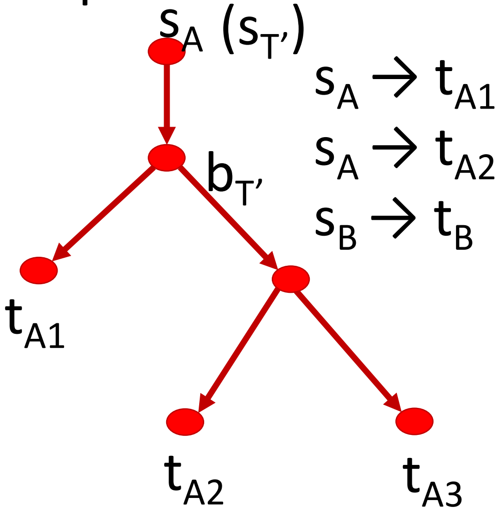
Backup slides

Example: Shared link detection



Ground truth

Step 1:

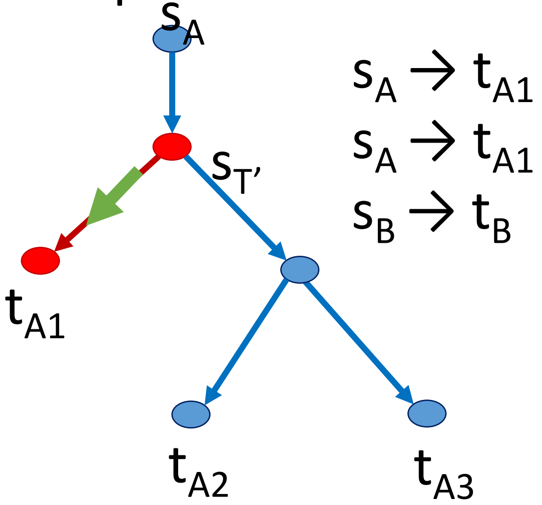


$s_A \rightarrow t_{A1}$

$s_A \rightarrow t_{A2}$

$s_B \rightarrow t_B$

Step 2:

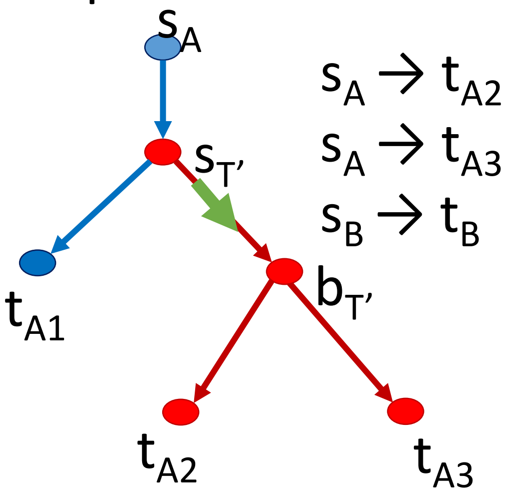


$s_A \rightarrow t_{A1}$

$s_A \rightarrow t_{A1}$

$s_B \rightarrow t_B$

Step 3:

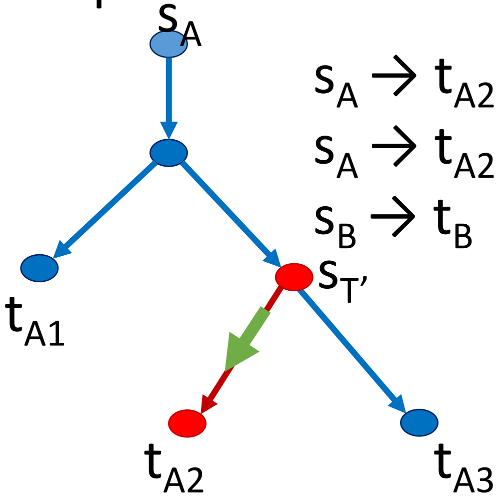


$s_A \rightarrow t_{A2}$

$s_A \rightarrow t_{A3}$

$s_B \rightarrow t_B$

Step 4:

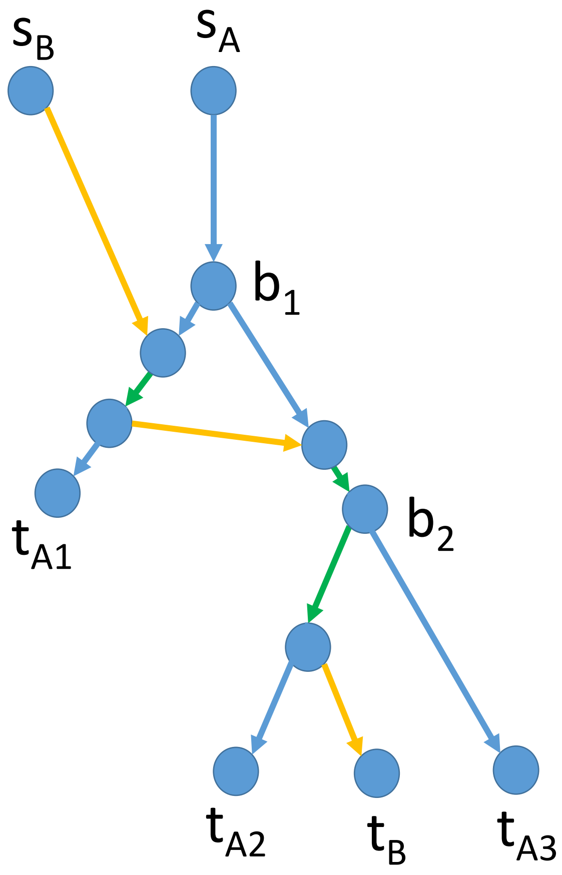


$s_A \rightarrow t_{A2}$

$s_A \rightarrow t_{A2}$

$s_B \rightarrow t_B$

Parameter Estimation



Ground truth

Top-down: One queue at a time

Attack rate

parameter

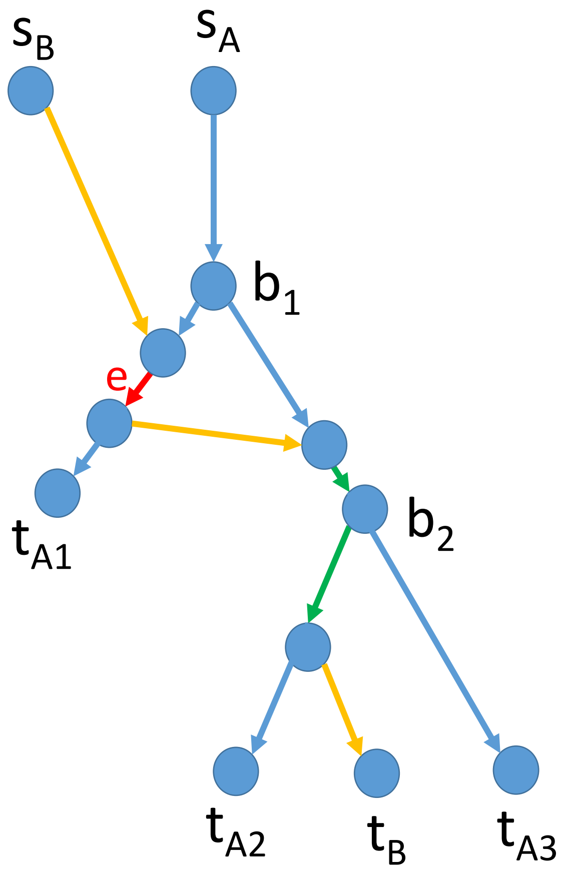
M/M/1: $d(r_e;\bar{\lambda}) = \dfrac{1}{r_e - \bar{\lambda}}$

M/D/1: $d(\lambda_e, \mu_e; \bar{\lambda}) = \dfrac{2\mu_e - \lambda_e - \bar{\lambda}}{2\mu_e(\mu_e - \lambda_e - \bar{\lambda})}$

G/G/1: $d(\lambda_e, \mu_e, \sigma_{ae}, \sigma_{se}; \bar{\lambda}) \approx$

$$\frac{1}{2\mu_e}\,\frac{\lambda_e + \bar{\lambda}}{\mu_e - \lambda_e - \bar{\lambda}}\left(\sigma_{ae}^2(\lambda_e + \bar{\lambda})^2 + \sigma_{se}^2\mu_e^2\right) + \frac{1}{\mu_e}$$

Parameter Estimation



Ground truth

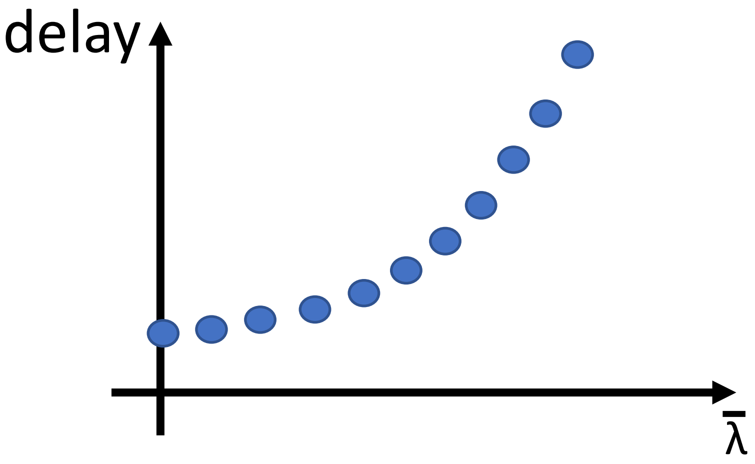
M/M/1: $d(r_e;\bar{\lambda}) = \frac{1}{r_e - \bar{\lambda}}$

Send probes s_A->t_{A1} with rate $\bar{\lambda}$ = 0, …, r/2

Measure delay of path s_B->t_B

$$c_{\mathcal{T}^*} + \frac{1}{r_e - \bar{\lambda}}$$

Constant



Theorem:

Accurate delay

& enough dimension

⇓

Accurate parameter