Internet routing measurements: from theory to practice

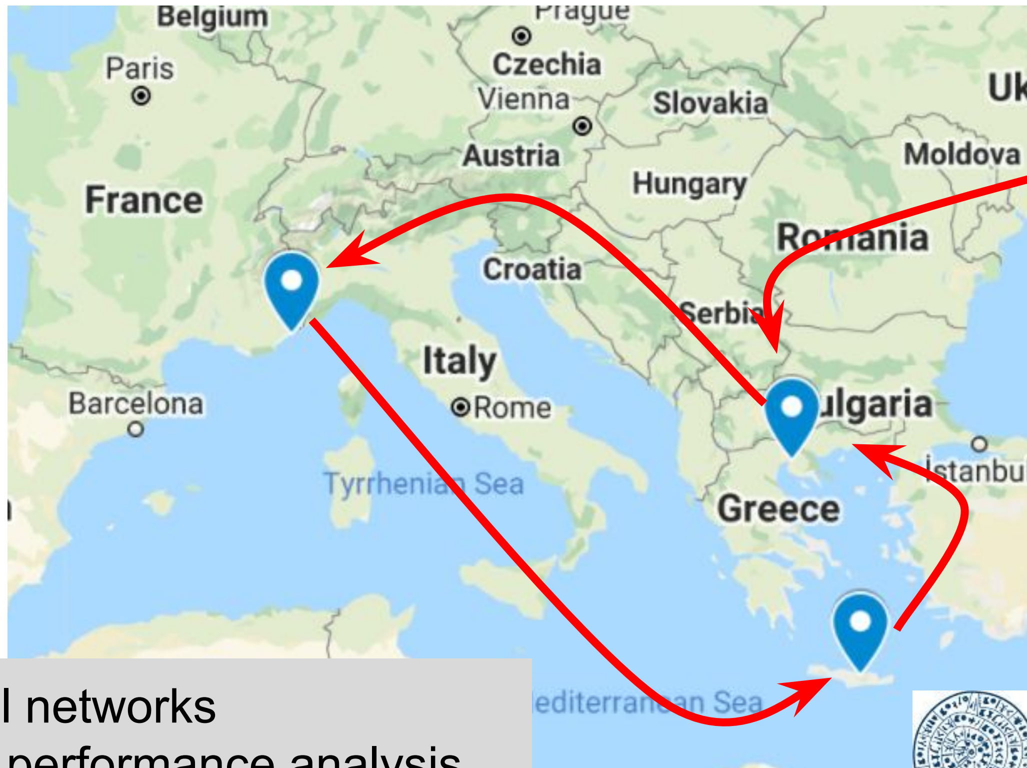
Pavlos Sermpezis

DataLab (datalab.csd.auth.gr),
Informatics Dept.,
Aristotle University of Thessaloniki

Short bio

ECE @ AUTH
Thessaloniki
(2005-2010)

PhD@EURECOM
Sophia-Antipolis
(2011-2015)



CSD @ AUTH
(2020-now)

INSTITUTE OF COMPUTER SCIENCE

FORTH, Heraklion
(2015-2019)

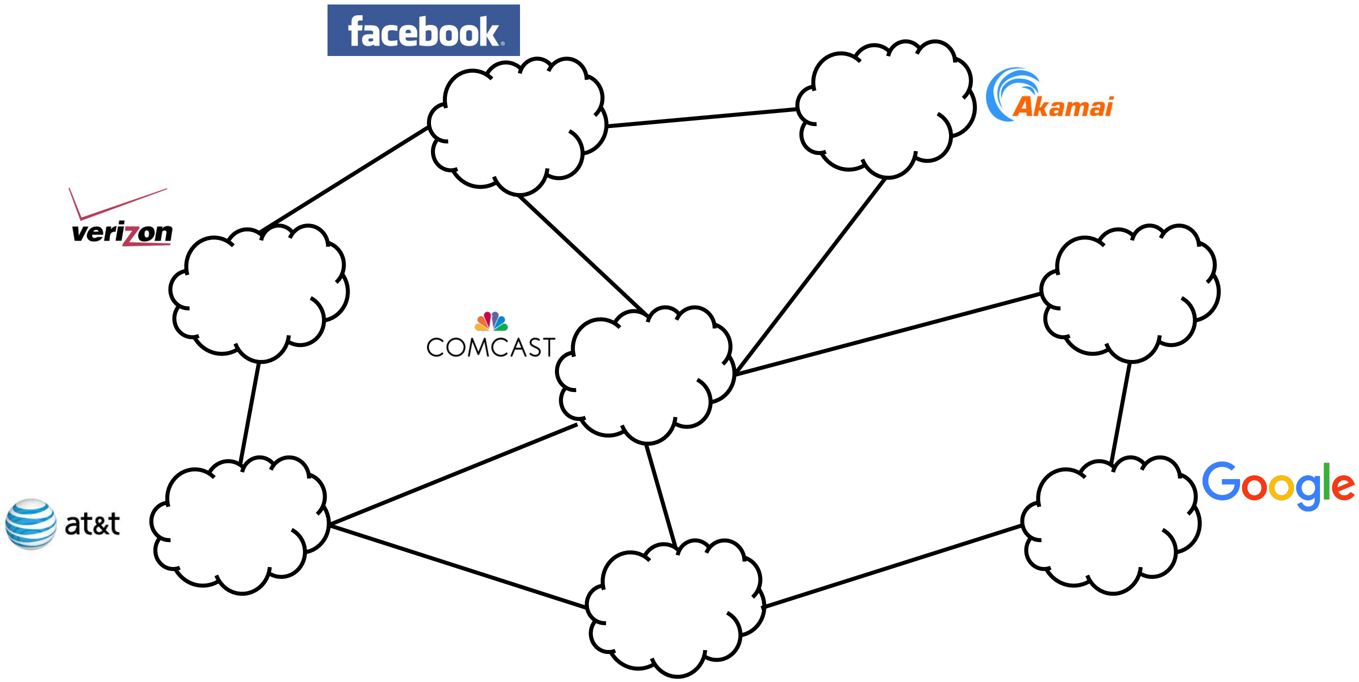
- mobile & social networks
    - modeling, performance analysis
- Internet routing & security
    - measurements, systems
- data science
    - web data, ML/AI research

The Internet

The Internet (routing) is a… ?

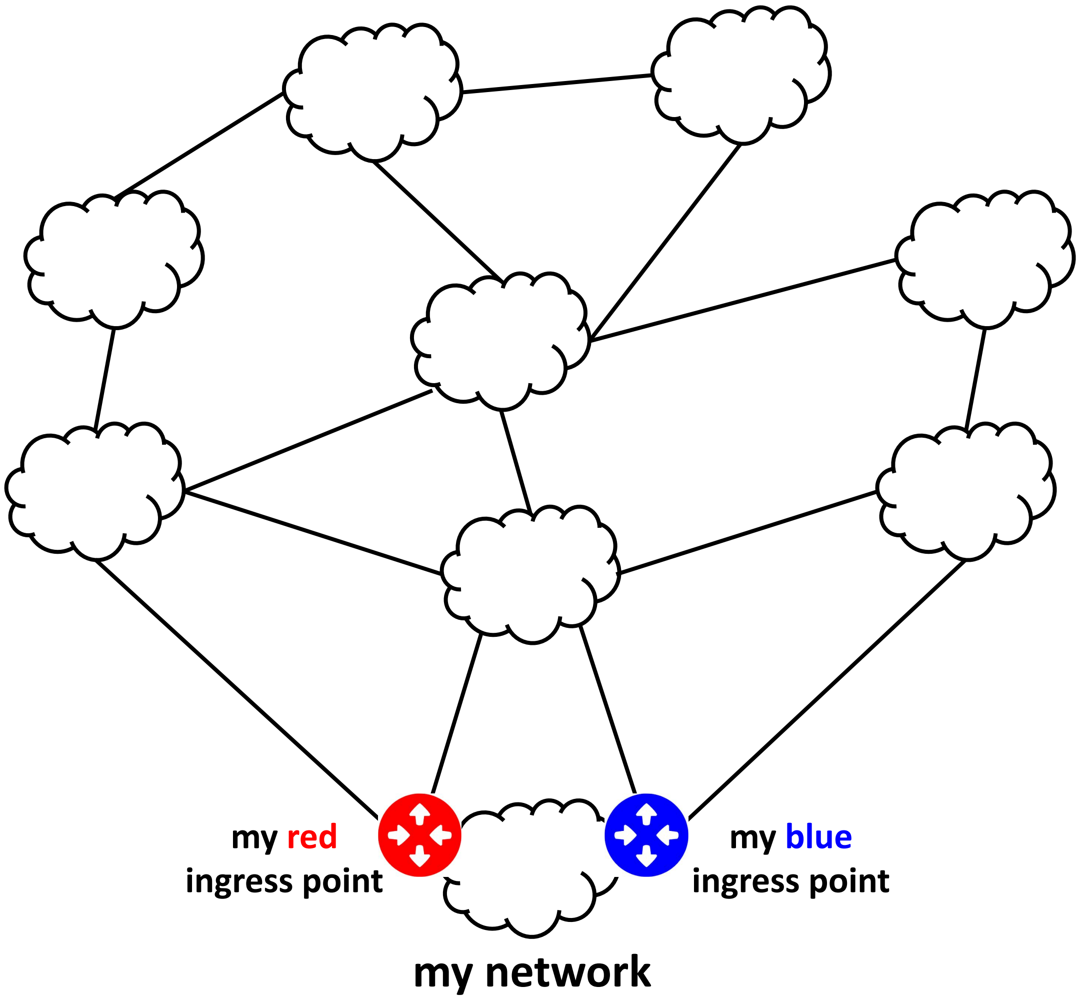
A. mesh
B. mess
C. well modeled and analyzed system

The Internet

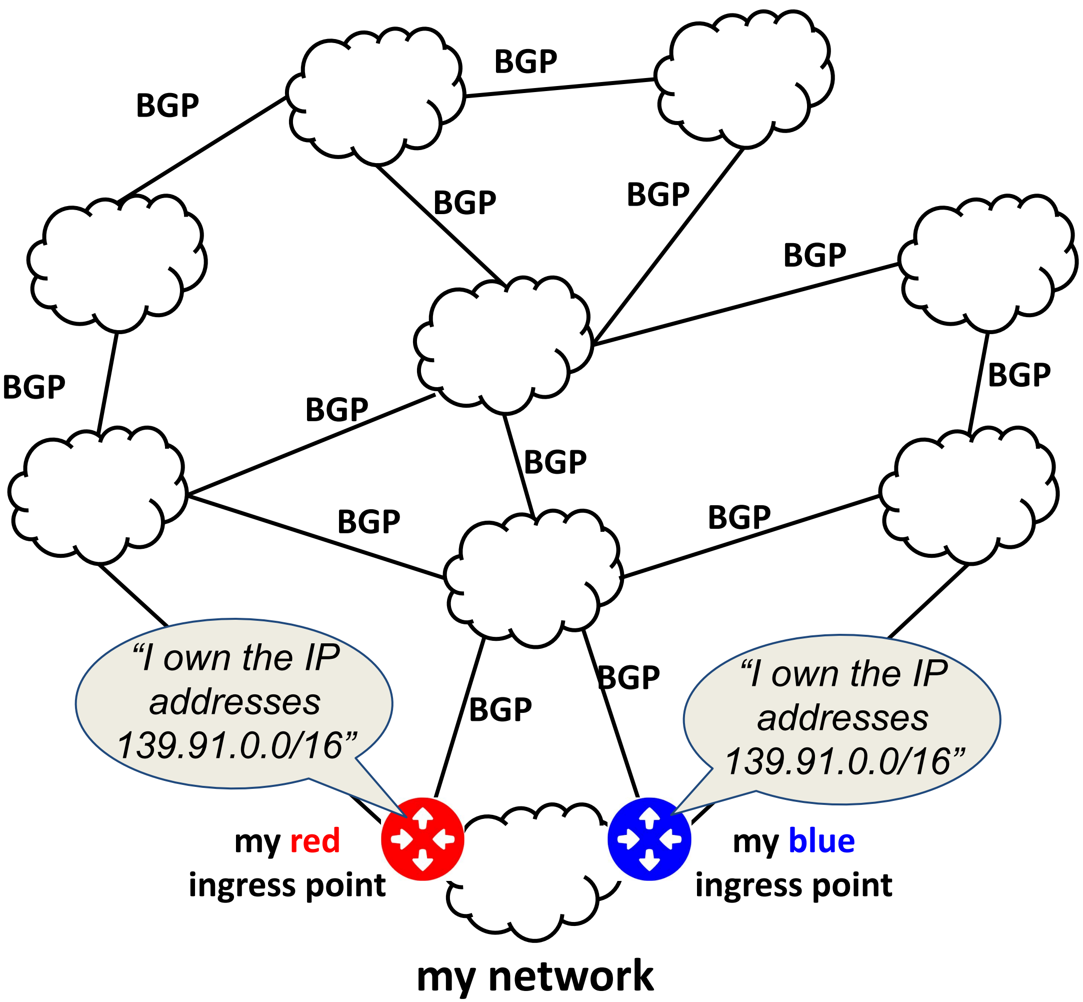


- The Internet is a network of networks or "**Autonomous Systems (AS)**"
- ***~70k*** ASes
- ***~450k*** AS-AS links ("peering links")

The Internet

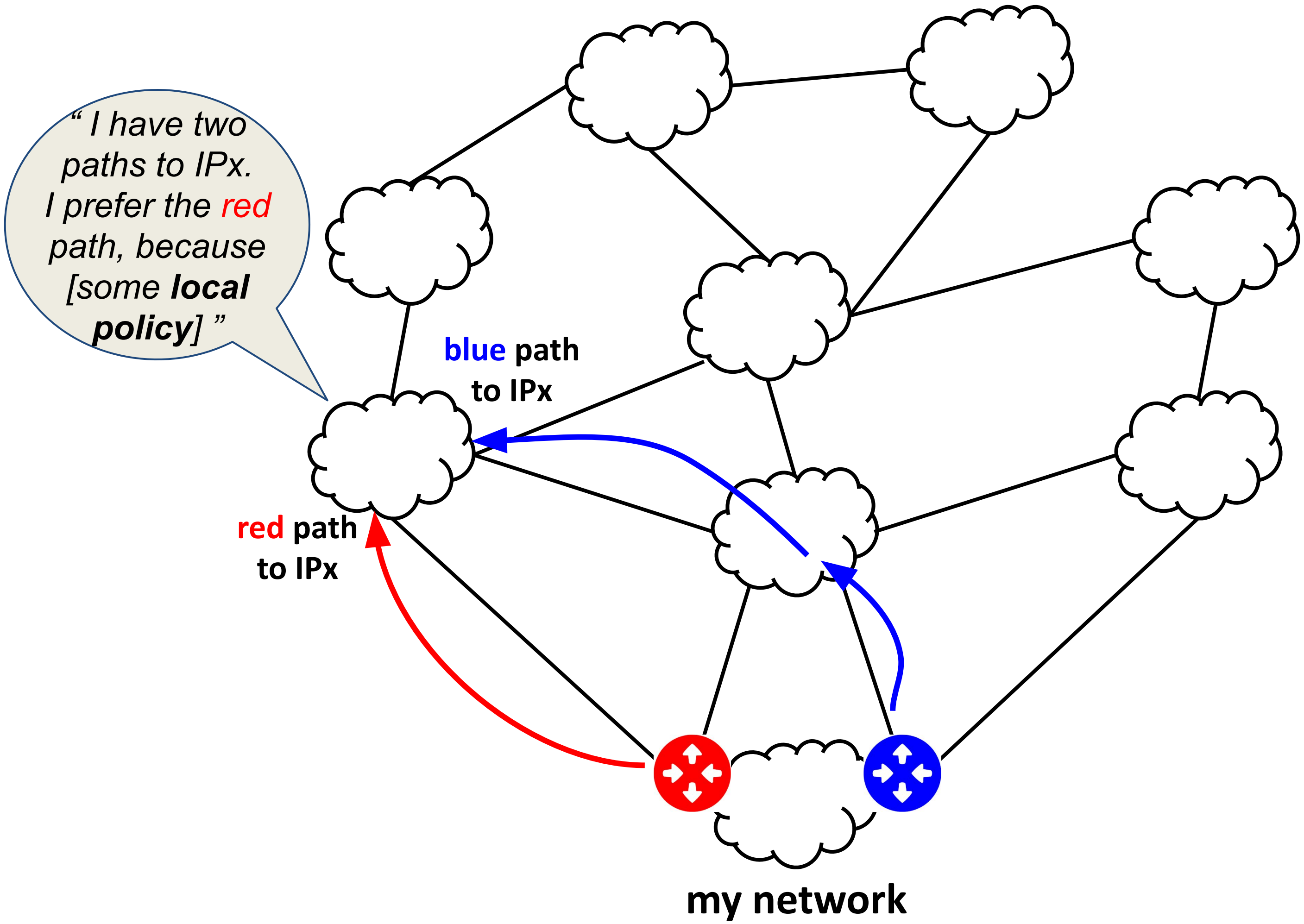


Internet routing

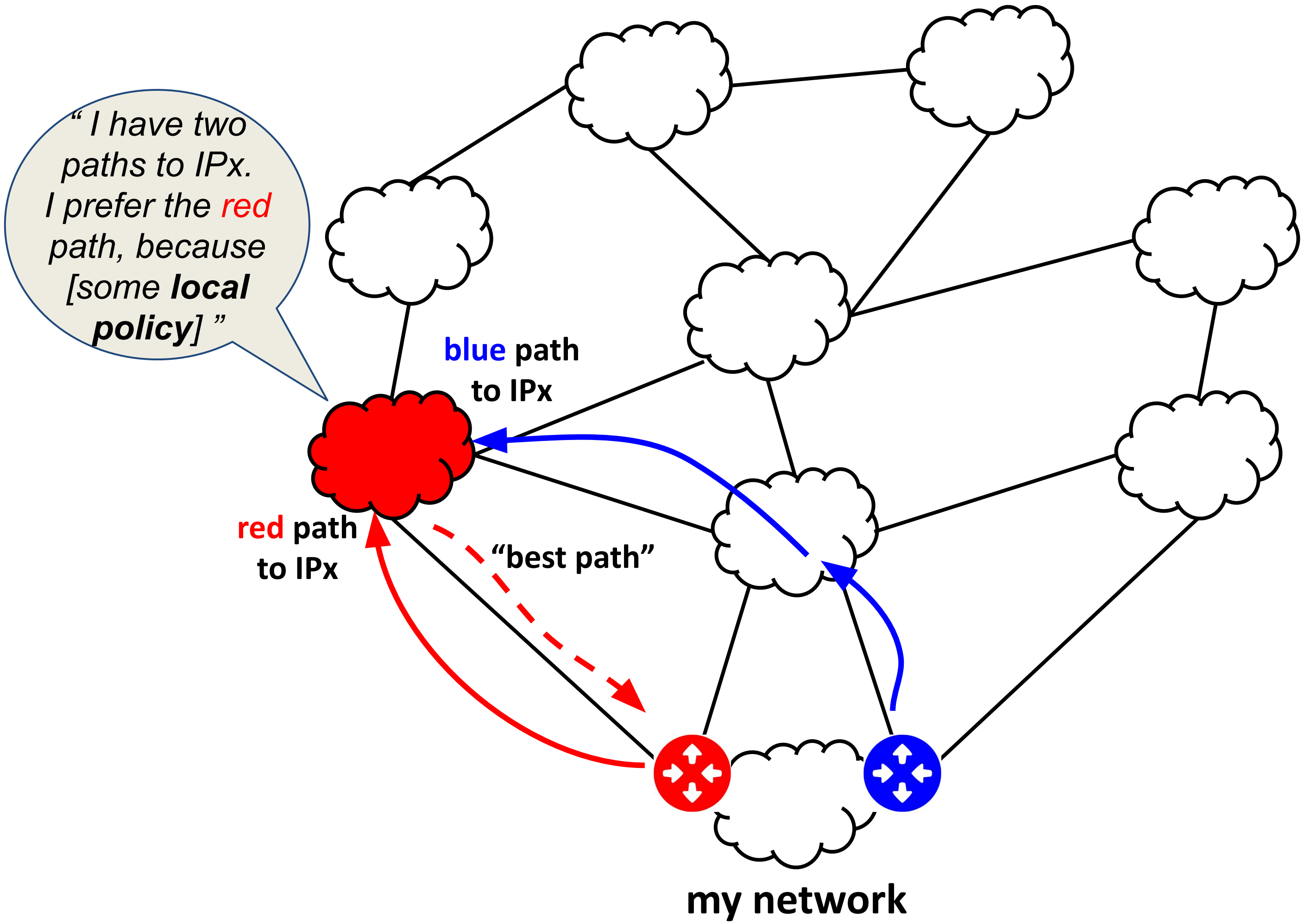


<u>Internet routing:</u>
Border Gateway Protocol (BGP)

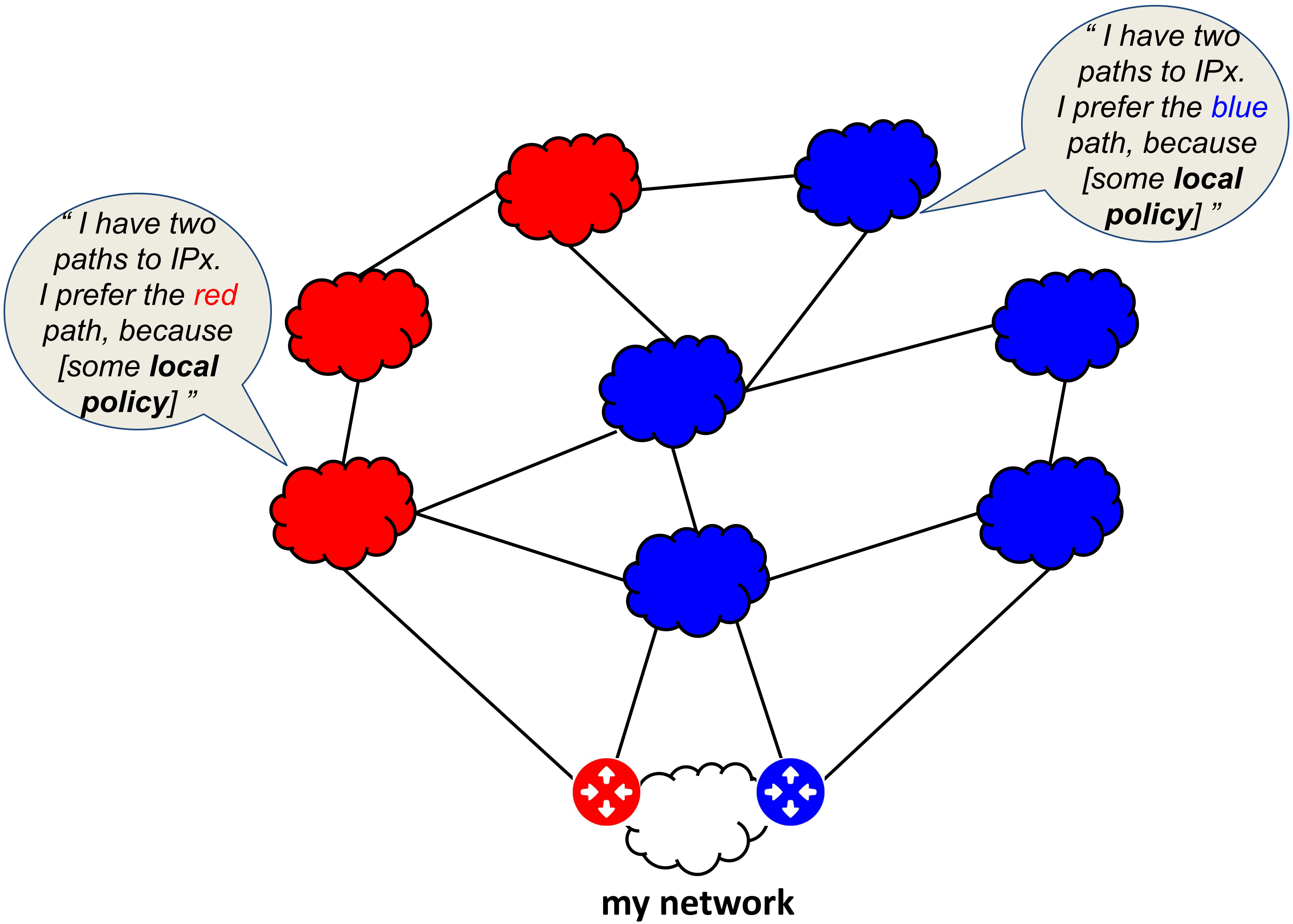
Internet routing



Internet routing



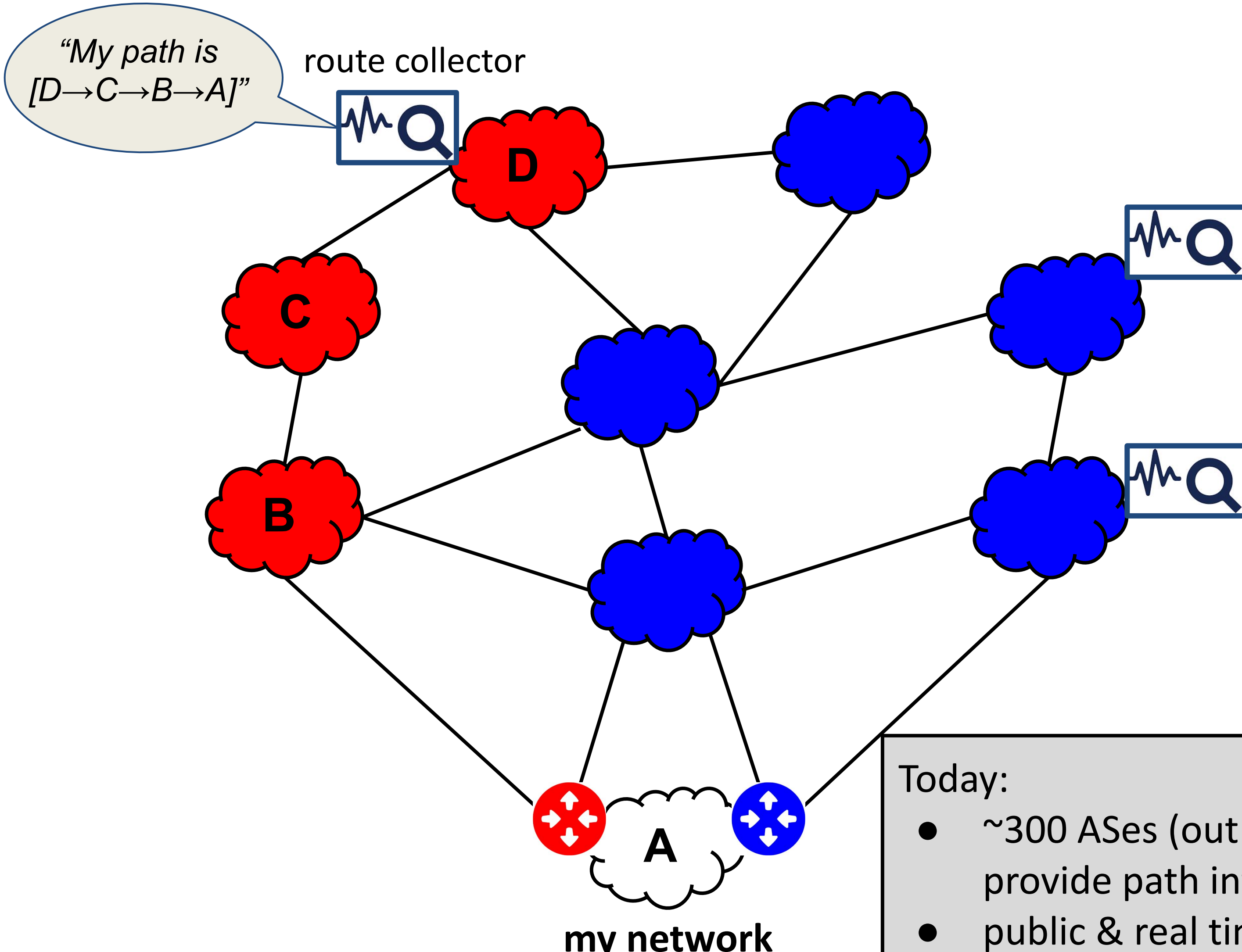
Internet routing



Internet routing measurements

	Characteristics		
	Information type	***Measurement type***	***Vantage points***
● **BGP route collectors**	*path*	*passive (monitoring)*	*public monitors*
● **Pings**	*reachability*	*active*	*public monitors & local (my network)*
● **Traceroutes**	*reachability & path*	*active*	*public monitors & local (my network)*

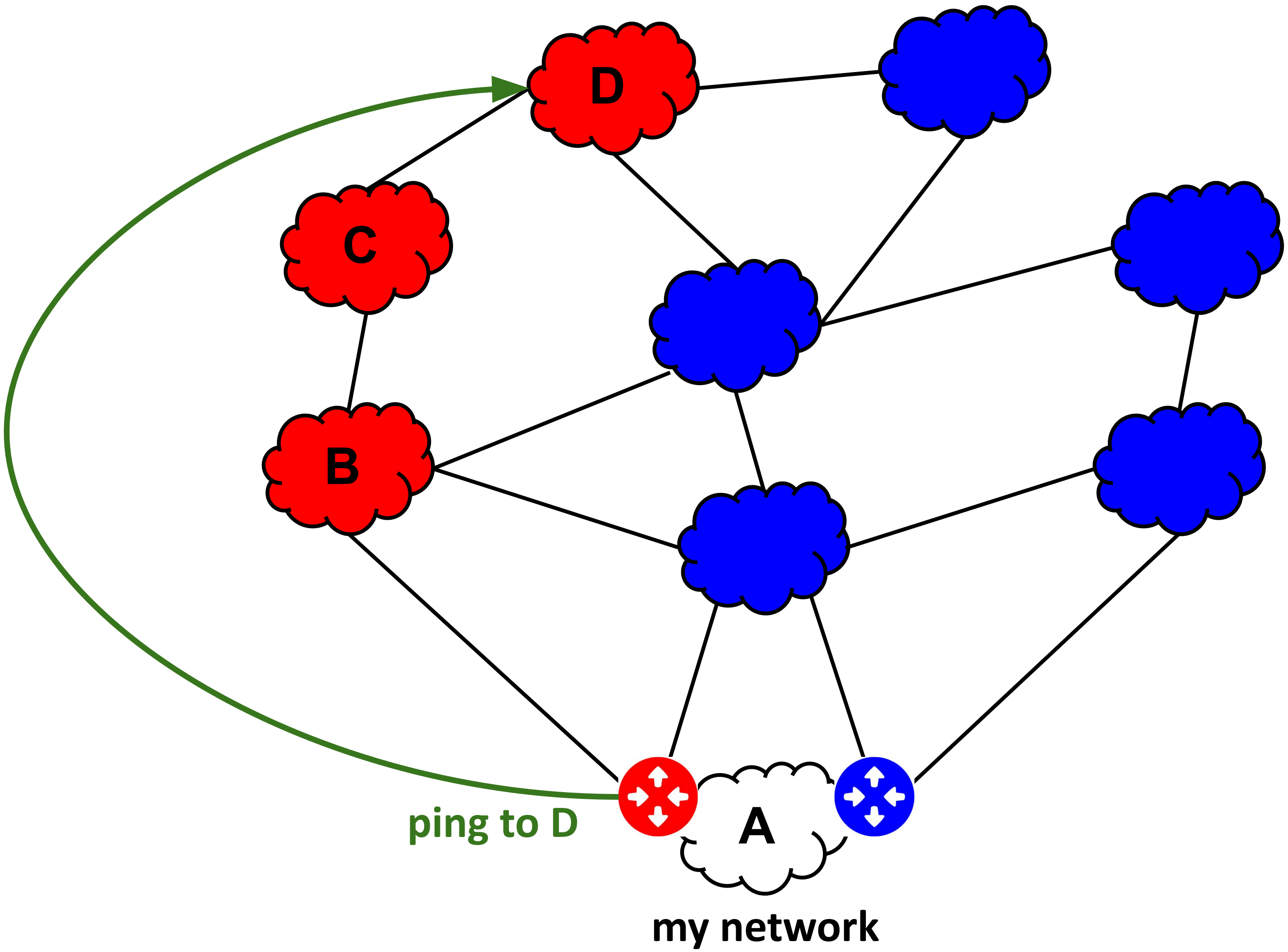
BGP route collectors



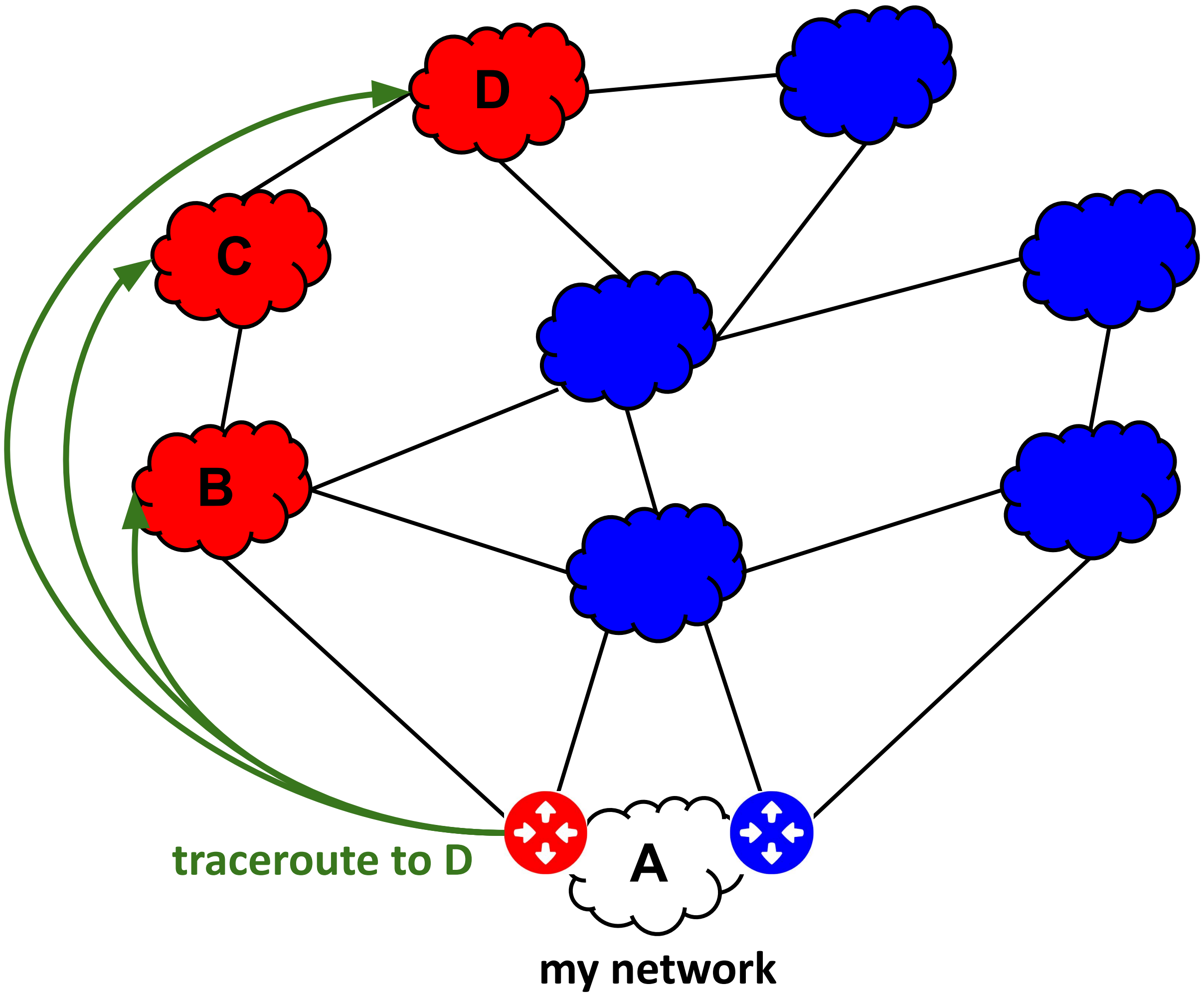
Today:

- ~300 ASes (out of all ~70k ASes) provide path information
- public & real time

Ping measurements



Traceroute measurements

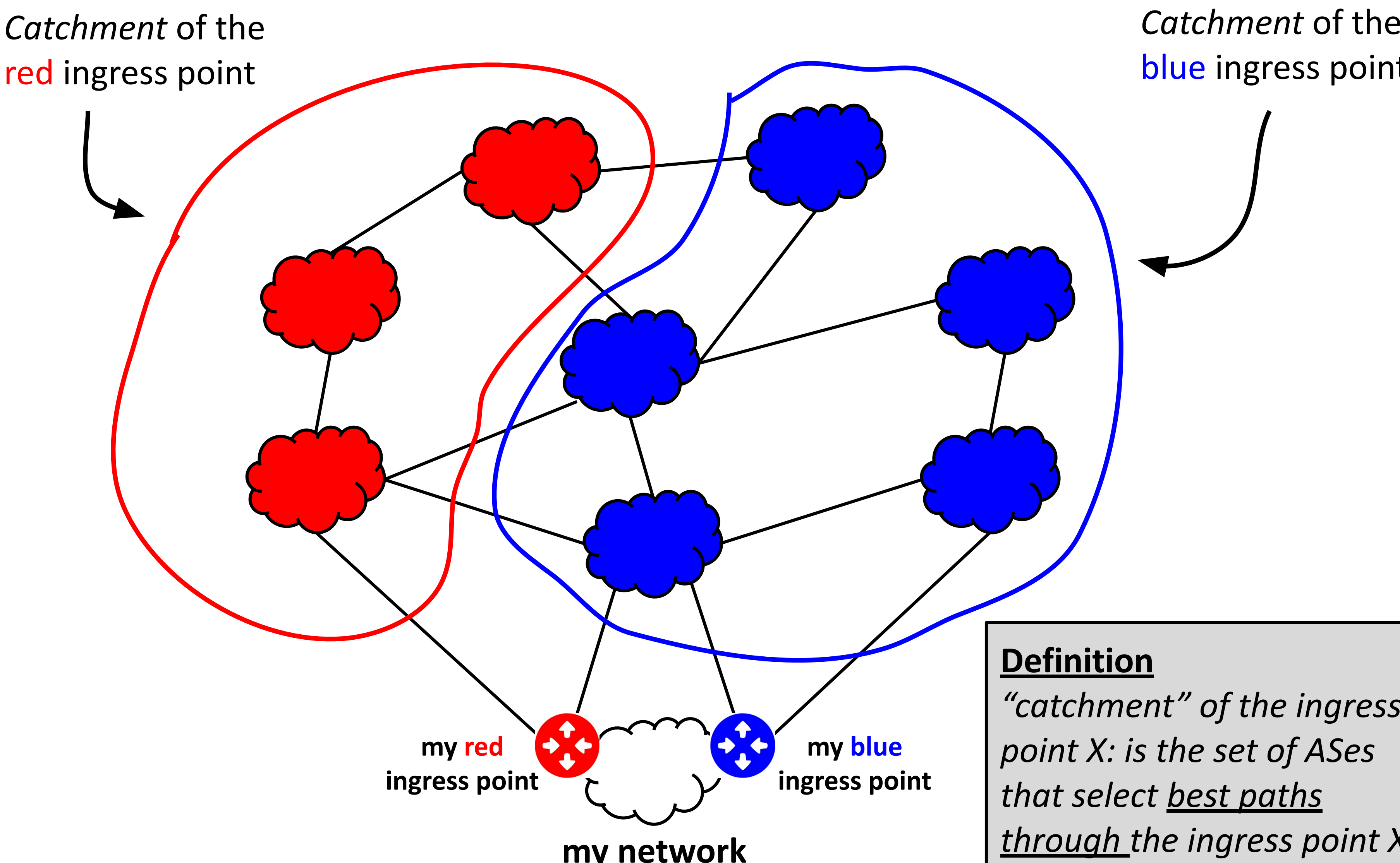


Use cases / problems

1. **Network management (“catchment” inference)**

2. **Network security (BGP prefix hijacking)**

Catchment



Definition

"catchment" of the ingress point X: is the set of ASes that select ***best paths through*** *the ingress point X*

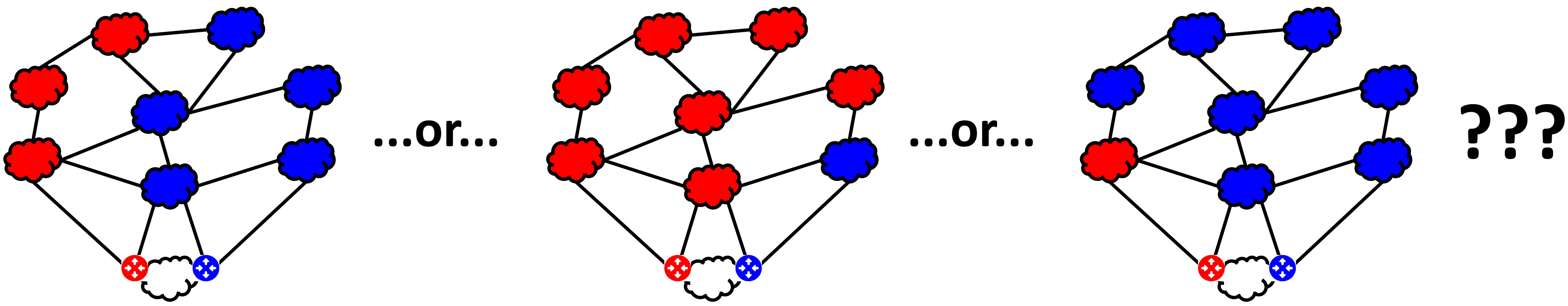
Importance of catchment

Knowing / inferring / predicting the catchment is important for:

- **load balancing & traffic engineering**
  - *e.g., "how to announce my IP addresses through BGP so that each of the ingress points A and B receives 50% of the incoming traffic?"*
- **network planning & resource allocation**
  - *e.g., "where to deploy/connect a new ingress point so that it attracts the incoming traffic from US-based ASes?"*
- **resilience analysis**
  - *e.g., "how would a BGP hijack by AS_x or failure of a link Y affect my traffic?"*
- **etc.**

Problem: What is my “catchment”?

A fundamental problem in Internet routing operations:
A network cannot fully ***control*** or even ***know*** exactly the catchment of its ingress points!



Why? Because the catchment depends on the...

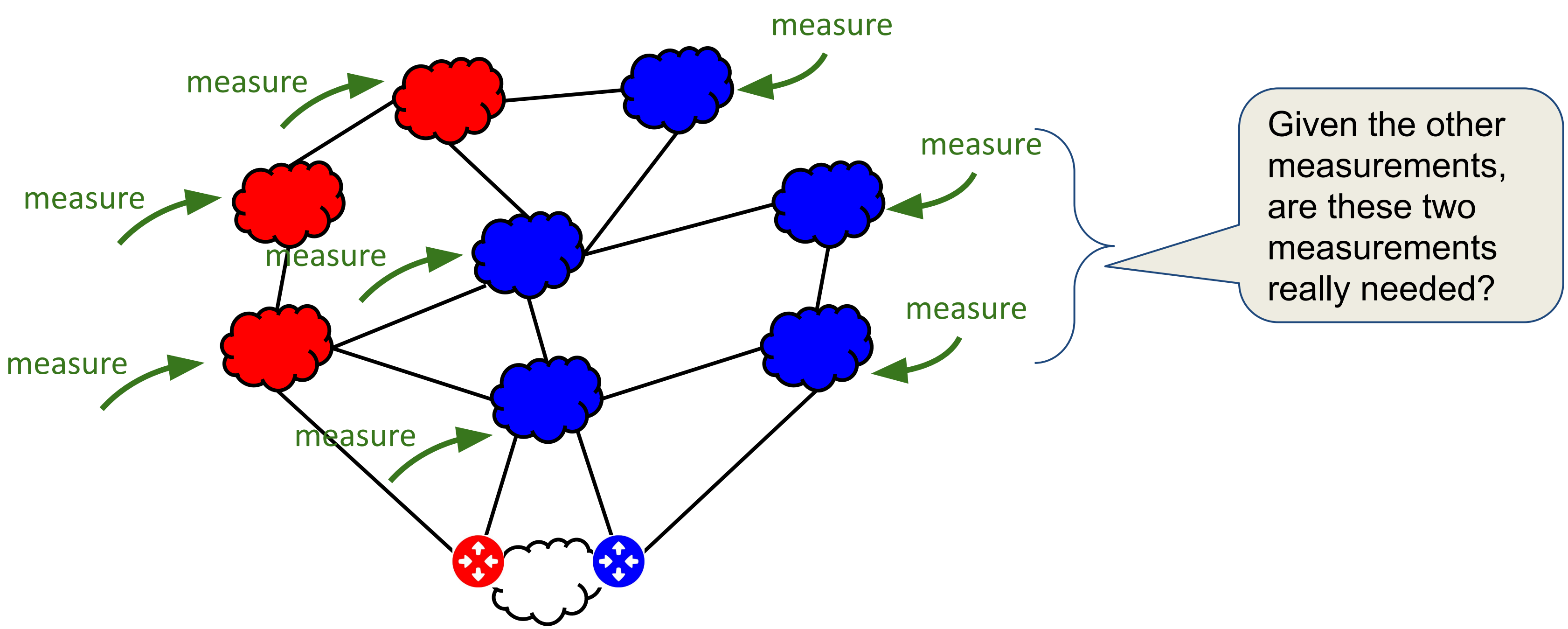
- routing configuration (set of ingress points, BGP announcement fields, etc.)
- local routing policies of all ASes → which are not accurately known
- topology & BGP dynamics → which are highly complex

Existing approaches

What do network operators do today to estimate / infer / predict the catchment?

Measurements!

- existing methodologies: measure all (~70k) networks → naive, inefficient, non-scalable



Can we do better?

- **Goals**
  - infer catchment with less (or no) measurements
  - propose efficient measurement methodologies
  - prediction even for non existing deployments

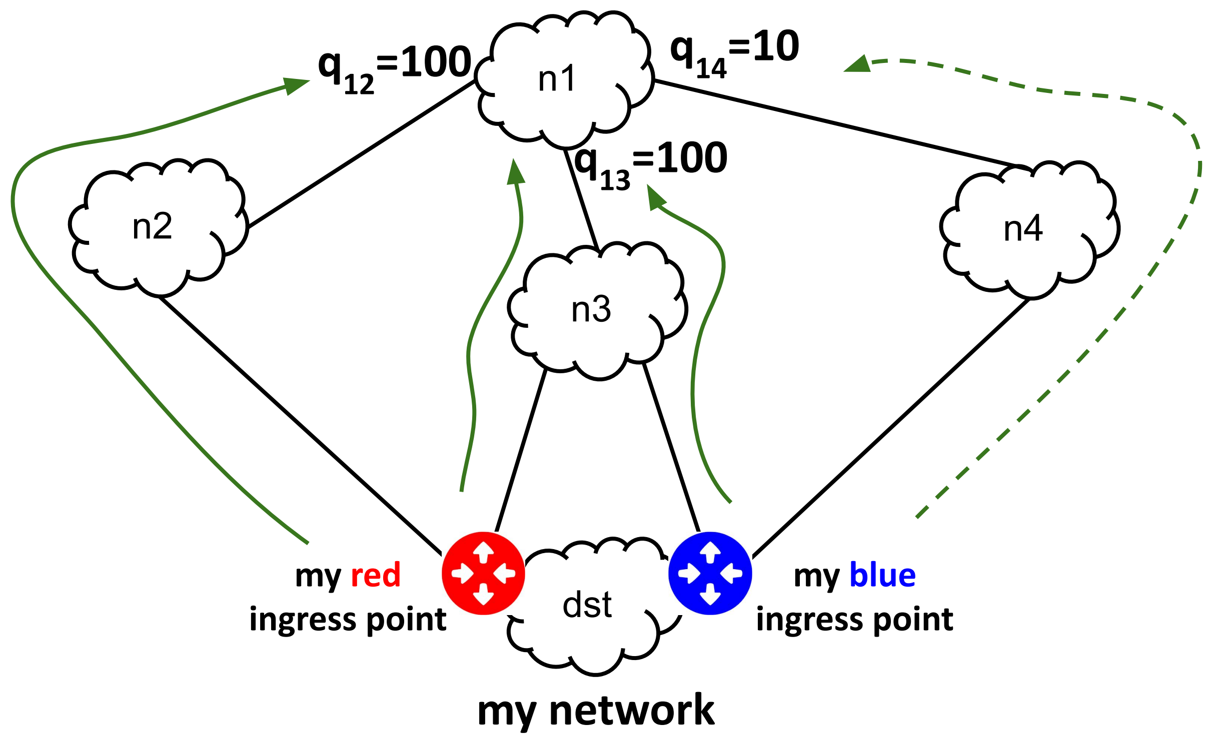
- **How?** exploit information:
  - Topology: ASes and links → we (mostly) know it
  - Routing policies → we have only partial/inaccurate information
  - Local policies (i.e., “my network” / ingress points) → we know it

more complete & accurate data → more informative our inference

*Pavlos Sermpezis and Vasileios Kotronis,. "**Inferring Catchment in Internet Routing**", ACM SIGMETRICS, 2019.*

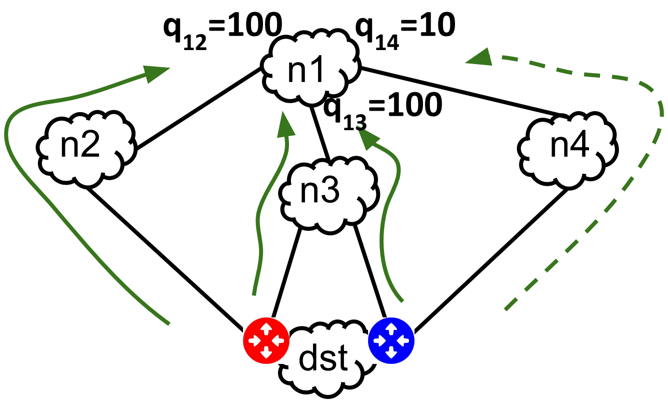
Let’s start: all paths, eligible paths, best paths

- **all possible BGP paths of n1**: *[dst→ … → n1]*
- **eligible paths of n1**: *[dst→n2→n1]* and *[dst→n3→n1]*
  - *[dst→n4→n1]* is ***not*** eligible, because $q_{12}, q_{13} > q_{14}$ (i.e., not preferred as best path)
- **best path of n1** ?
  - best path == catchment inference

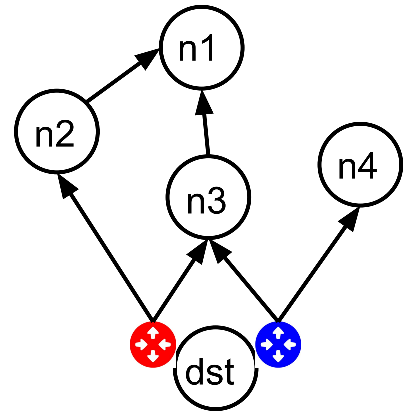


Inference, step 0: build the R-graph

- The **R-graph** is a directed acyclic graph (DAG) that encodes all the eligible paths

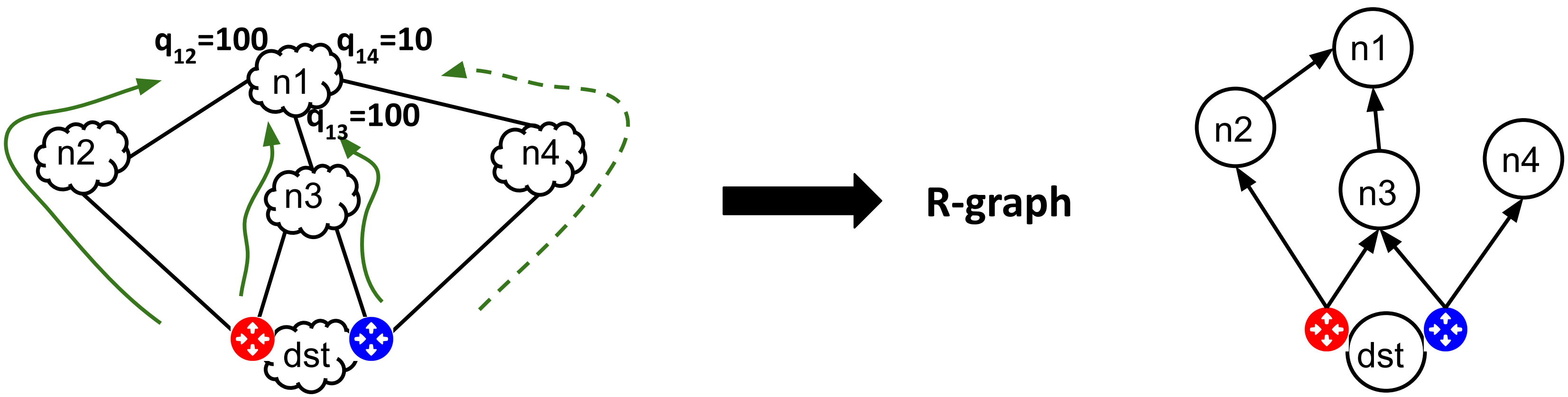


R-graph



Inference, step 0: build the R-graph

- The **R-graph** is a directed acyclic graph (DAG) that encodes all the eligible paths



Algorithm 1 (R-graph construction)

- Simulate BGP, arbitrarily break ties (this randomness does *not* affect the R-graph construction!)
- FOR each node, add incoming edges from all neighbors: (i) from which a path is learned, <u>and</u> (ii) have the highest local preference

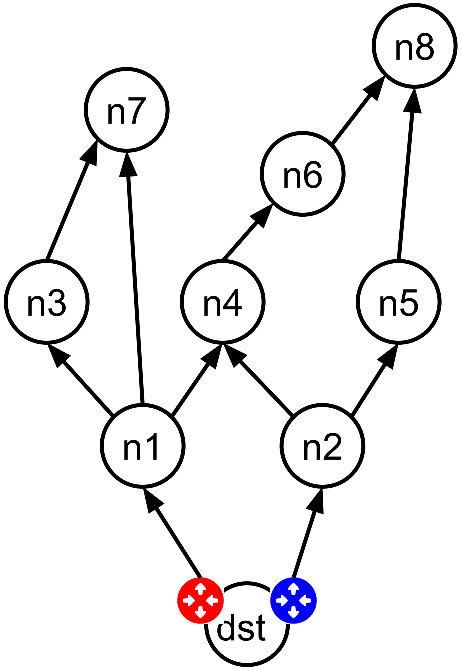
Theorem 1

A path $p_{i \to dst}$ = [*dst* → …→ *i*] is an eligible path <u>if and only if</u> it can be constructed by starting from node *dst* and following a sequence of directed edges in the R-graph until reaching node *i*

Certain inference

- The R-graph …
  - … encodes all the eligible paths
  - … enables catchment inference
    (i.e., infer through which ingress point each node *i* routes its traffic to node *dst*)

Algorithm 2 (certain inference on the R-graph)

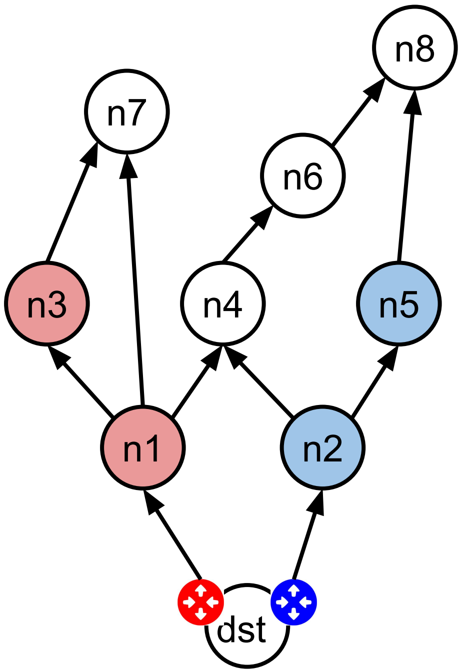


Certain inference

- The R-graph …
  - … encodes all the eligible paths
  - … enables catchment inference
    (i.e., infer through which ingress point each node *i* routes its traffic to node *dst*)

Algorithm 2 (certain inference on the R-graph)

- IF only one eligible path → inference
  e.g,. nodes n1, n2, n3, n5

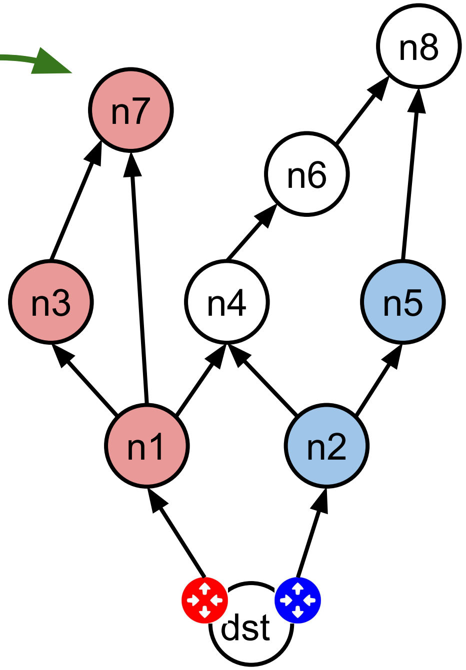


Certain inference

- The R-graph …
  - … encodes all the eligible paths
  - … enables catchment inference
    (i.e., infer through which ingress point each node *i* routes its traffic to node *dst*)

Algorithm 2 (certain inference on the R-graph)

- IF <u>only one eligible path</u> → inference
  e.g,. nodes n1, n2, n3, n5
- IF <u>multiple eligible paths</u>
  - IF all paths through the same ingress point → inference
    e.g., node n7

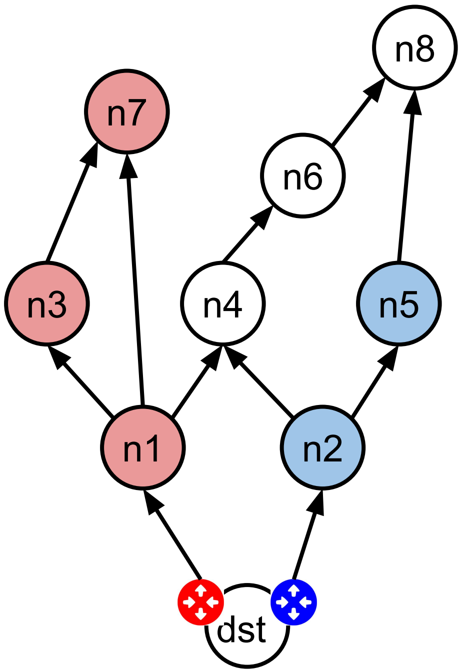


Certain inference

- The R-graph …
  - … encodes all the eligible paths
  - … enables catchment inference
    (i.e., infer through which ingress point each node *i* routes its traffic to node *dst*)

Algorithm 2 (certain inference on the R-graph)

- IF <u>only one eligible path</u> → inference
  e.g,. nodes n1, n2, n3, n5
- IF <u>multiple eligible paths</u>
  - IF all paths through the same ingress point → inference
    e.g., node n7
  - ELSE → no certain inference
    e.g., nodes n4, n6, n8



Certain inference

- The R-graph …
  - … encodes all the eligible paths
  - … enables catchment inference
    (i.e., infer through which ingress point each node *i* routes its traffic to node *dst*)

→ certain inference is possible for many nodes even when multiple paths are eligible

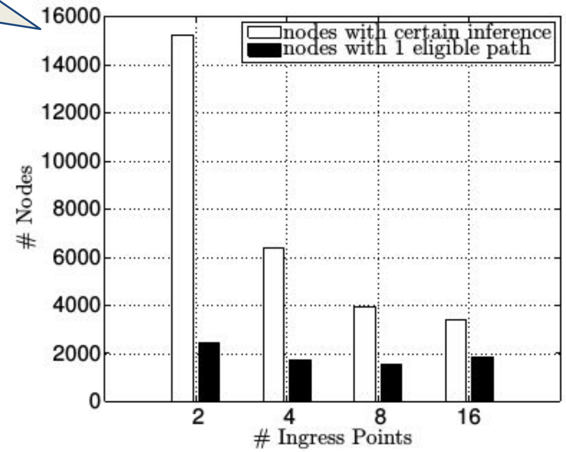
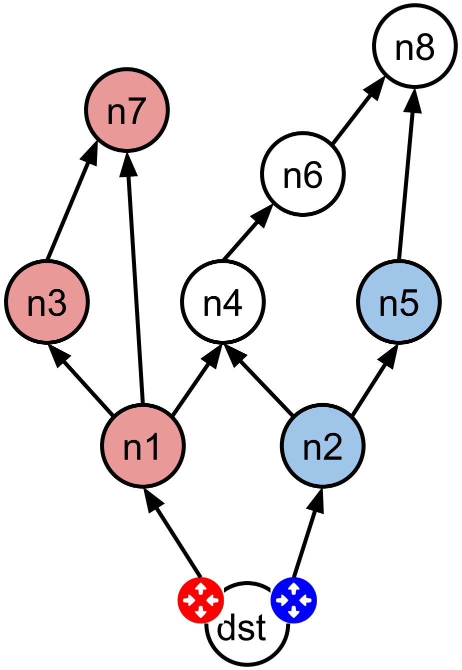
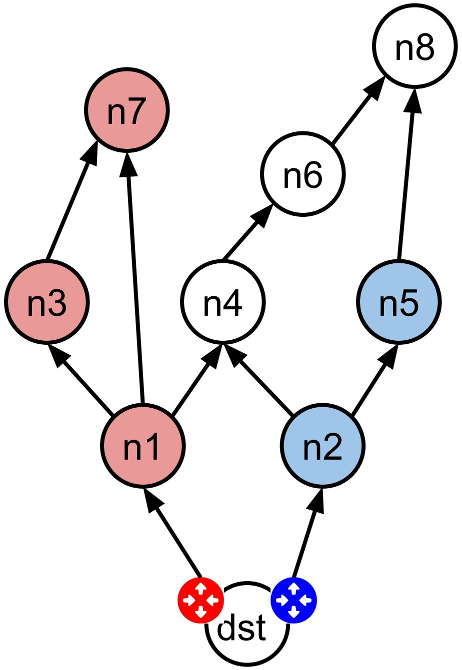


Figure 1
#nodes with certain inference (white) vs.
#nodes with only 1 eligible path (black)



Probabilistic inference

- ***"route probability"*** $\boldsymbol{\pi_i(m)}$: probability the best path of node ***i*** to be through the ingress point ***m***
  - *e.g.,* $\pi_{n4}(\text{red}) = ?$ *,* $\pi_{n4}(\text{blue}) = ?$



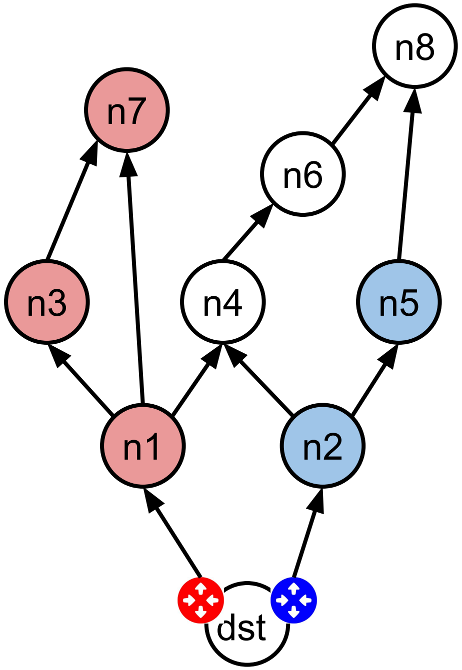
Probabilistic inference

- ***"route probability" $\pi_i(m)$***: probability the best path of node ***i*** to be through the ingress point ***m***
    - *e.g., π_{n4}(red) = ? , π_{n4}(blue) = ?*

Algorithm 3 (probabilistic inference on the R-graph)

- FOR *node i* in topological_sort(R-graph)
    - calculate probability from parent nodes $\mathbf{C_i}$

$$\pi_i(m) = \sum_{j \in C_i} \pi_j(m) \cdot \frac{1}{|C_i|}$$



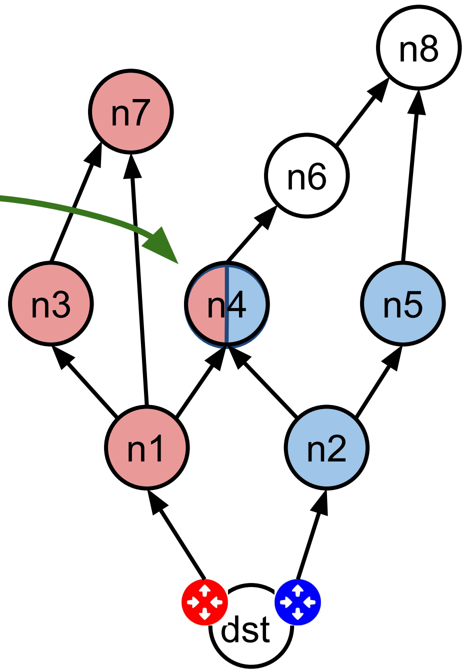
Probabilistic inference

- ***"route probability" $\pi_i(m)$***: probability the best path of node ***i*** to be through the ingress point ***m***
  - *e.g.,* π_{n4}(*red*) = ? , π_{n4}(*blue*) = ?

Algorithm 3 (probabilistic inference on the R-graph)

- FOR *node i* in topological_sort(R-graph)
  - calculate probability from parent nodes $\mathbf{C_i}$

$$\pi_i(m) = \sum_{j \in C_i} \pi_j(m) \cdot \frac{1}{|C_i|}$$



e.g.,

- *node n4:* π_{n4}(*red*) = 1/2 , π_{n4}(*blue*) = 1/2

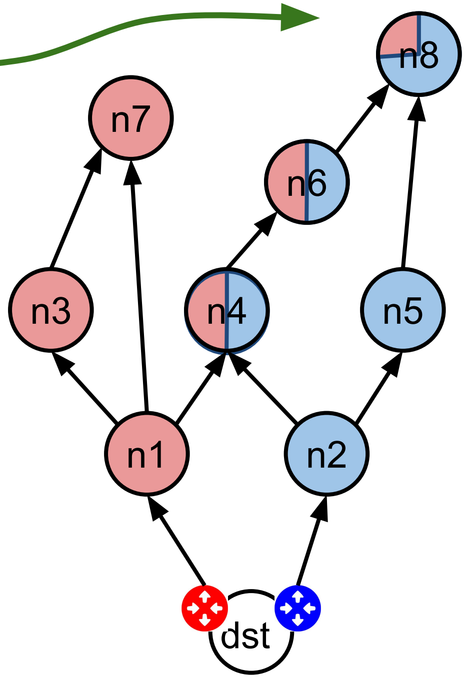
Probabilistic inference

- ***"route probability"* $\pi_i(m)$**: probability the best path of node ***i*** to be through the ingress point ***m***
  - *e.g., π_{n4}(red) = ? , π_{n4}(blue) = ?*

Algorithm 3 (probabilistic inference on the R-graph)
- FOR *node i* in topological_sort(R-graph)
  - calculate probability from parent nodes $\mathbf{C}_i$

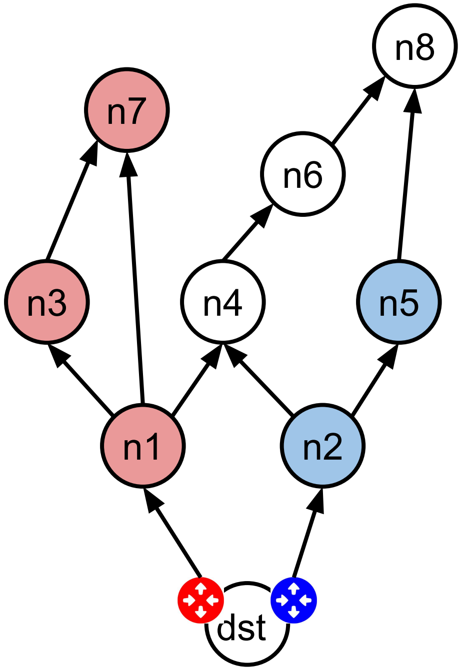
$$\pi_i(m) = \sum_{j \in C_i} \pi_j(m) \cdot \frac{1}{|C_i|}$$

e.g.,
- *node n4: π_{n4}(red) = 1/2 , π_{n4}(blue) = 1/2*
- *node n6: π_{n6}(red) = 1/2 , π_{n6}(blue) = 1/2*
- *node n8: π_{n8}(red) = 1/4 , π_{n8}(blue) = 3/4*



Inference under oracles

- oracle = measurement (e.g., BGP route collector, traceroute, etc.)
- Goal: measure a set of nodes **S** ⟶ infer the catchment for **S'** ⊇ **S** ?
- more efficient measurement strategies

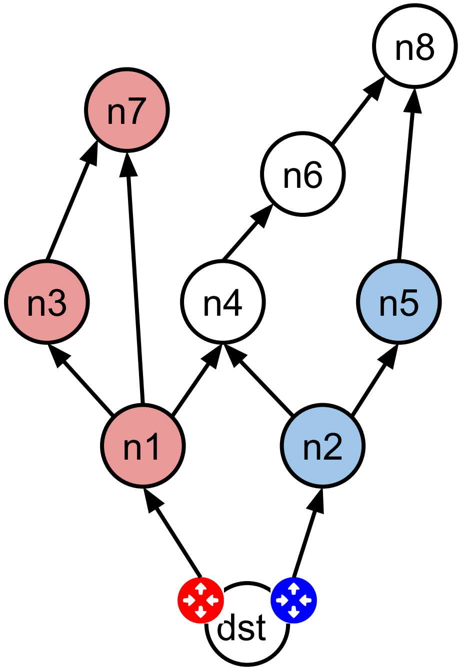


Inference under oracles

- oracle = measurement (e.g., BGP route collector, traceroute, etc.)
- Goal: <u>measure</u> a set of nodes **S** → <u>infer</u> the catchment for **S'** ⊇ **S** ?
- more efficient measurement strategies

Algorithm 4 (enhance inference under oracles)

- for a set of nodes **S**, measure ("infer from oracle") their routes



Inference under oracles

- oracle = measurement (e.g., BGP route collector, traceroute, etc.)
- Goal: measure a set of nodes **S** ⟶ infer the catchment for **S'** ⊇ **S** ?
- more efficient measurement strategies

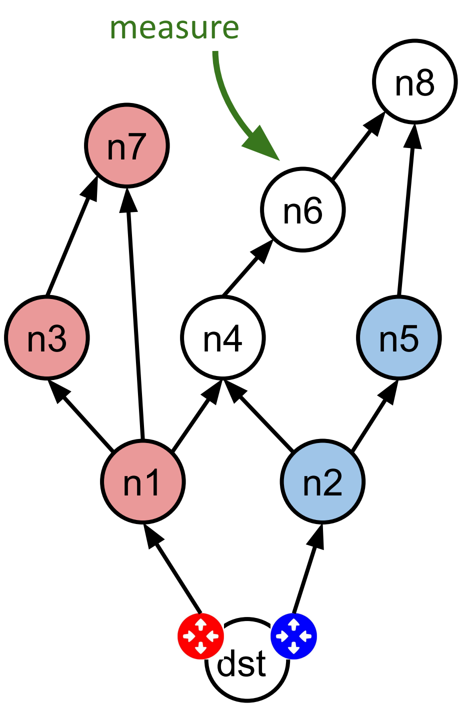
Algorithm 4 (enhance inference under oracles)

- for a set of nodes **S**, measure (“infer from oracle”) their routes
- Iteratively
  - infer the routes of the children-nodes of nodes in **S**

    *e.g., measure node **n6***

    *if **n6**=blue ⟶ **n8**=blue*

    *if **n6**=red ⟶ no inference for **n8***

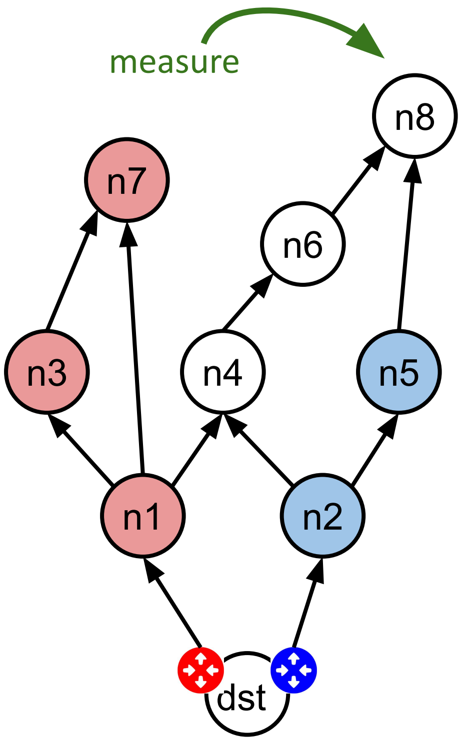


Inference under oracles

- oracle = measurement (e.g., BGP route collector, traceroute, etc.)
- Goal: <u>measure</u> a set of nodes **S** $\longrightarrow$ <u>infer</u> the catchment for **S'** $\supseteq$ **S** ?
- more efficient measurement strategies

Algorithm 4 (enhance inference under oracles)

- for a set of nodes **S**, measure (“infer from oracle”) their routes
- Iteratively
  - infer the routes of the <u>children-nodes</u> of nodes in **S**
    *e.g., measure node **n6***
    *if **n6**=blue $\longrightarrow$ **n8**=blue*
    *if **n6**=red $\longrightarrow$ no inference for **n8***
  - infer the routes of the <u>parent-nodes</u> of nodes in **S**
    *e.g., measure node **n8***
    *if **n8**=red $\longrightarrow$ no inference for **n6**=red*
    *if **n8**=blue $\longrightarrow$ no inference for **n6***

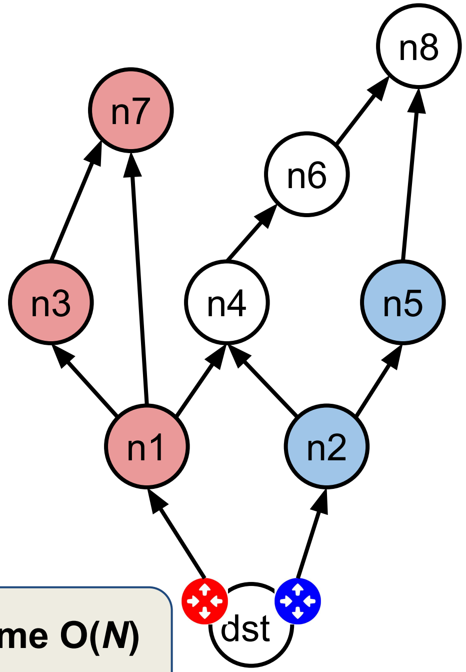


Inference under oracles

- oracle = measurement (e.g., BGP route collector, traceroute, etc.)
- Goal: measure a set of nodes **S** ⟶ infer the catchment for **S' ⊇ S** ?
- more efficient measurement strategies

Algorithm 4 (enhance inference under oracles)

- for a set of nodes **S**, measure ("infer from oracle") their routes
- Iteratively
  - infer the routes of the children-nodes of nodes in **S**
    *e.g., measure node **n6***
    *if **n6**=blue ⟶ **n8**=blue*
    *if **n6**=red ⟶ no inference for **n8***
  - infer the routes of the parent-nodes of nodes in **S**
    *e.g., measure node **n8***
    *if **n8**=red ⟶ no inference for **n6**=red*
    *if **n8**=blue ⟶ no inference for **n6***



- Updating the certain inference after measurements → **polynomial time O(*N*)**
- Updating the probabilistic inference after measurements → **NP-hard**

Efficient measurement strategies

- Optimal measurement strategy → NP-hard problem
  - combinatorial; without "nice properties" (e.g., submodularity)
  - even updating probabilities under oracles is NP-hard
    ("belief updating" in non-polytree Bayesian Networks; reduction to SAT)
- Heuristic (greedy) algorithm
  - at each step select measurement that increases most the certain inference
  - update probabilities only from *forward* belief propagation

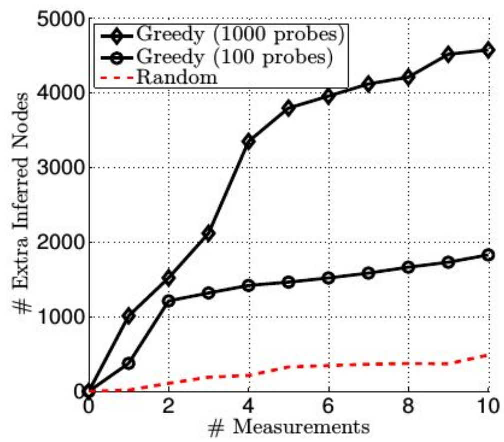
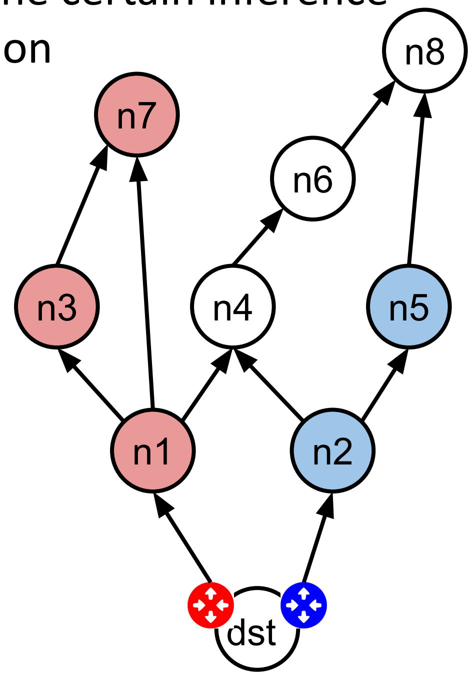


Figure 3
R-graph based heuristic (black) vs.
random measurements (red)



More results (in the paper)

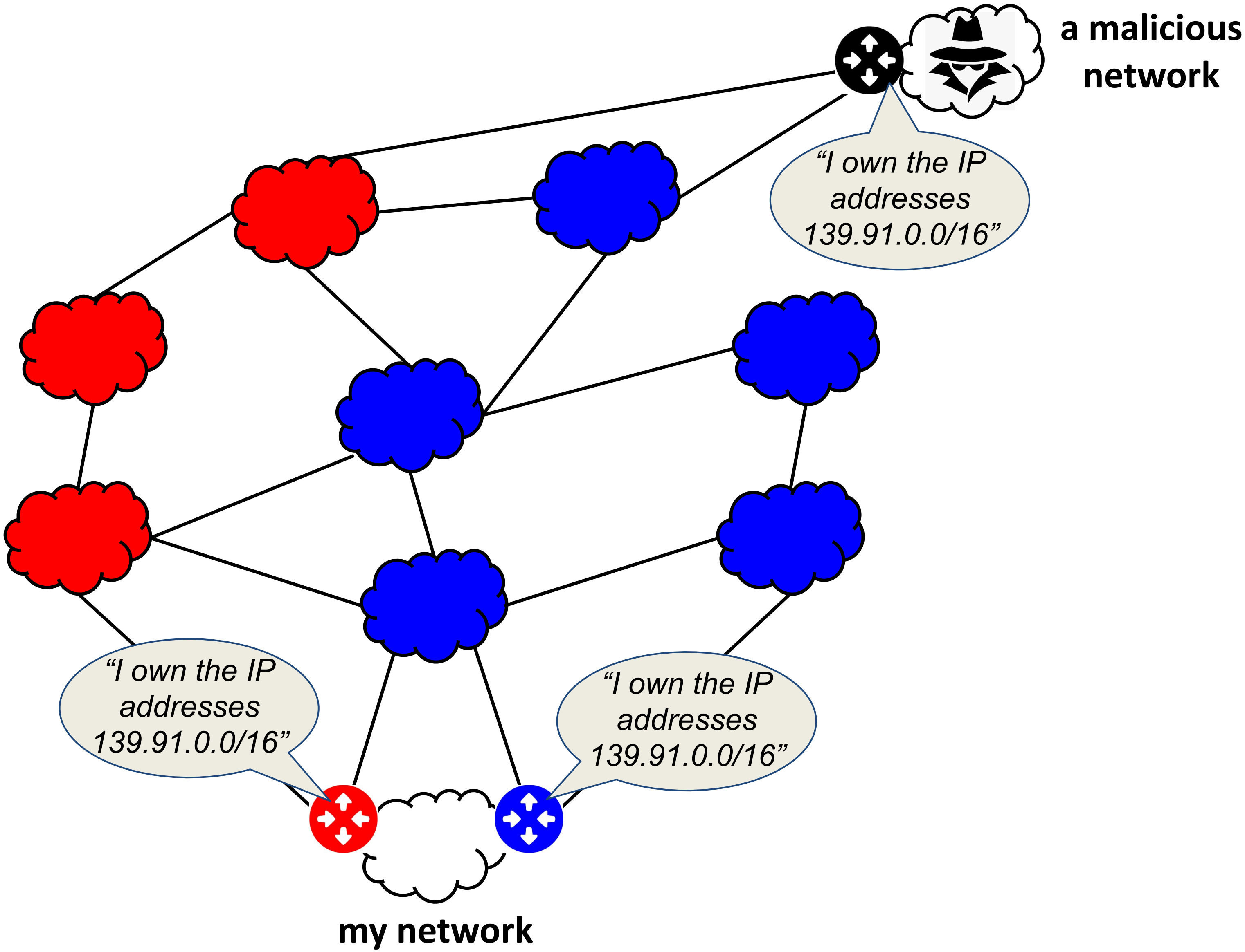
- **rich information from inference**
  - upper/lower bounds, mean values, important links/nodes/policies
- **completeness of inference**
- **efficiency of existing monitoring infrastructure**
  - RouteViews, RIPE, LGs, etc.
- **real experiments**
  - accuracy (vs. real-world measurements for IP anycasting)
  - traffic engineering (vs. real-world peering selection)

Use cases / problems

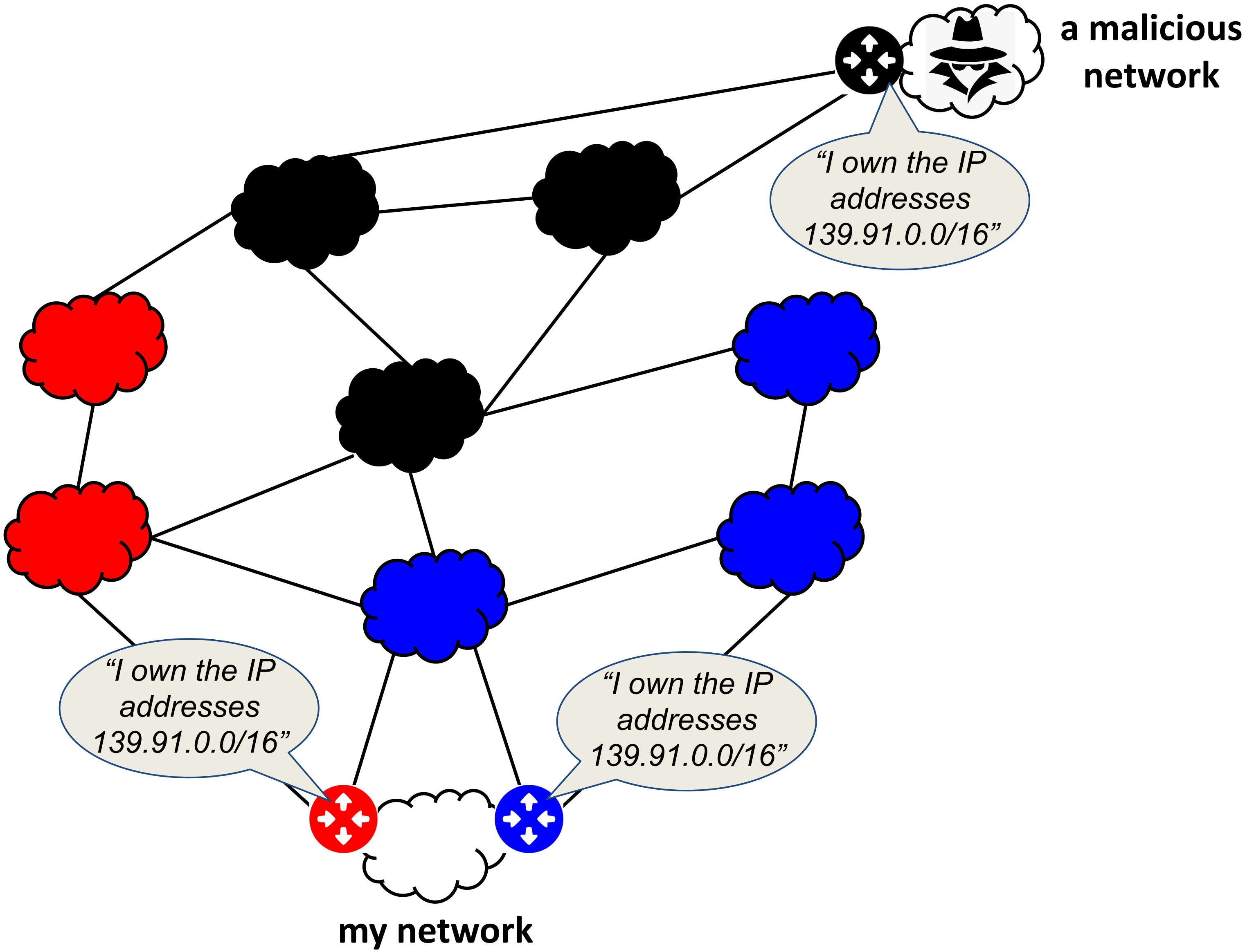
1. **Network management ("catchment" inference)**

2. **Network security (BGP prefix hijacking)**

BGP Prefix hijacking



BGP Prefix hijacking



Importance of BGP prefix hijacking

- **service outages & traffic interception**
  - *can last for hours*
  - *affect millions of users*
  - *can cost 100s of thousands of $$$ (or more) <u>per minute</u>*

- **~2500 (reported) prefix hijacking events in 2020**

- **no actual (proactive) defence**
  - *defences based upon detection & countermeasures*
  - *timely detection is very important*

BGP prefix hijacking

- **Detection**
  - *our contribution → near real time detection (within a few seconds; ~5sec.)*

*Pavlos Sermpezis, et al. "**ARTEMIS: Neutralizing BGP Hijacking within a Minute**", in ACM/IEEE Transactions on Networking (ToN), 2018.*

in collaboration with UCSD

- **Impact estimation**
  - *impact == catchment of the hijacker*
  - *our contribution → 1st formal study on hijack impact estimation*

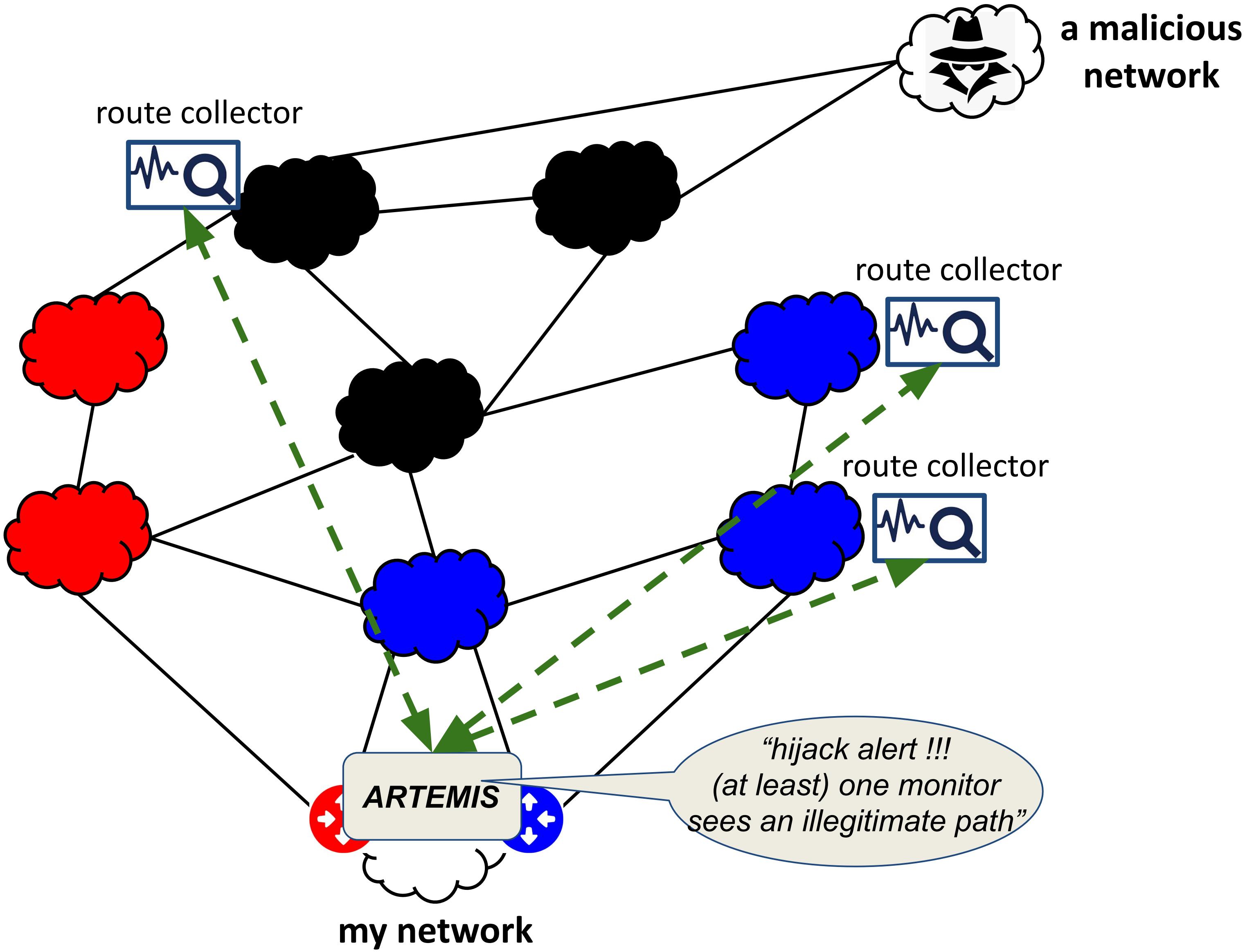
*Pavlos Sermpezis, et al. "**Estimating the Impact of BGP Prefix Hijacking**", in IFIP Networking conference, June 2021.*

Detection with ARTEMIS

- **“ARTEMIS”**
  - *real-time detection & automated mitigation system*

Pavlos Sermpezis, et al. "**ARTEMIS: Neutralizing BGP Hijacking within a Minute**", in ACM/IEEE Transactions on Networking (ToN), 2018.

Detection with ARTEMIS



Detection with ARTEMIS

- **"ARTEMIS"**
  - *real-time detection & automated mitigation system*
  - *open-source software https://bgpartemis.org/*
  - *deployed in operational networks*
    - *Internet2, AMS-IX, ForthNet, etc.*

- **Public monitoring infrastructure (BGP route collectors)**
  - became real-time only recently (3-4 years)
  - rich information (will become richer with BMP protocol)
  - enables novel applications/methods (research & commercial)

*Pavlos Sermpezis, et al. "**ARTEMIS: Neutralizing BGP Hijacking within a Minute**", in ACM/IEEE Transactions on Networking (ToN), 2018.*

Impact estimation

- **Goal: estimate the impact of a (detected) hijacking event**
  - *detection (at least one network infected) vs.*
    impact estimation (number of infected networks)
  - *desired characteristics: fast & accurate methodology*

- **How?**
  - Heavyweight methodology: measure all networks [NOW]
    - *e.g., ping all networks*
  - Lightweight methodology: measure some networks (sampling)
    - *e.g., estimate from BGP route collectors* [NOW?]
    - *e.g., ping some networks*

*Pavlos Sermpezis, et al. "**Estimating the Impact of BGP Prefix Hijacking**", in IFIP Networking conference, June 2021.*

Impact estimation

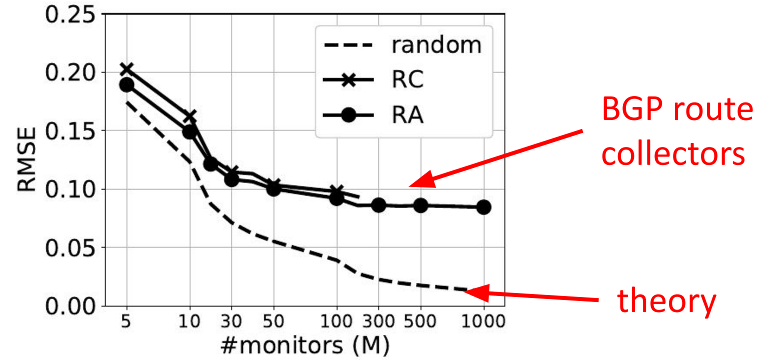
- **Sampling in theory…**
  - The estimation error (RMSE) decreases with the number of samples (M)

Theorem 1. *Under a randomly selected set of monitors $\mathcal{M}$, the bias and root mean square error of NIE are given by*

$$Bias_{NIE} = 0 \qquad RMSE_{NIE} = \frac{1}{\sqrt{M}} \cdot c_I$$

where $c_I = \int_0^1 \sqrt{I \cdot (1 - I)} \cdot f(I) \cdot dI$, *is a constant that depends on the impact distribution* $f(I)$.

- **Sampling in practice…**
  - with BGP route collectors



  - with ping measurements
    → high measurement failures (> *90%* non pingable IP addresses)
    → they end-up being less accurate than BGP route collectors (for ***p*** > *20%*)
    → we need ***p*** < *10%*

Theorem 2. *RMSE vs. failure probability* ***p*****.**

*Pavlos Sermpezis, et al. "**Estimating the Impact of BGP Prefix Hijacking**", in IFIP Networking conference, June 2021.*

(Practical) ping-based estimator

- **Ping-based estimator:**
  - Find “pingable” IP addresses for every AS [*ANT Lab’s IP hitlist*]
  - Ping multiple (N_{IP}) IP addresses per AS
  - If at least one ping reply from an AS → the AS is not affected by the hijack

N_{IP}	1	2	3	...	10
p	12.8%	4.2%	2.1%	...	0%
RMSE (M=100)	7.9%	4.7%	4.1%	...	3.9%

Key findings
- N_{IP}>=2 for low error
- no need for N_{IP}>3

Pavlos Sermpezis, et al. "**Estimating the Impact of BGP Prefix Hijacking**", in IFIP Networking conference, June 2021.

(Improved) BGP route collectors estimator

- BGP route collectors estimator
  - high error, why? → due to location bias, i.e., correlated measurements
  - how to improve? → decouple them! … with ML(?)

- **ML-based estimator:**
  - many features *(e.g., location information, routing policies, graph properties)* & ML models *(regression)* worked fine in simulations!
  - … but, in practice?
    - no labelled real datasets **:(**
    - we did real experiments in the Internet → ~20 labeled samples **:/**
    - most ML models did not train well in the few real experiments **:(**

Result: a *Linear (Ridge) Regressor* able to train well (20 samples!)
and achieve low estimation error (even in experiments!) **:)**

*Pavlos Sermpezis, et al. "**Estimating the Impact of BGP Prefix Hijacking**", in IFIP Networking conference, June 2021.*

ML/AI for networking: challenges & research directions

Data: networking has a lot of useful data, but...
- lack of labelled data
- lack of benchmark datasets
- heterogeneous data (topology, routing paths, link state, text in mailing lists(!), etc.)

Adoption: ML/AI techniques can be very efficient, but...
- network operators/admins are not data scientists
- lack of transparency (i.e., trust issues; critical tasks)

→ AutoML

→ Explainable AI (XAI)

Methods: general ML/AI methods may need to be adapted for networking problems...
- robustness
- cost function (high cost of errors)
- dynamic data
- capture network structure

→ Graph Neural Networks (GNN)

in collaboration with EPFL

Summarizing...

Measurements

- very important in Internet operations (due to complexity and limited modeling)
- today: many capabilities & opportunities (new technologies)
- current methodologies: large potential for improvement

Theory vs. practice

- Only-theory: does not work
- Only-practice: inefficient, suboptimal
- Mix theory & practice: develop theory → get insights → apply in practice

Future research directions

- Data-science & ML/AI for networking (*e.g., AutoML, Explainable AI, Graph ML)*
- Key for feasibility and adoption: involve network operators in research