JOINT NETWORK AND CONTENT ROUTING

OR, ON USING TRAFFIC ENGINEERING AT SERVICE LEVEL

Vytautas Valancius, ***Bharath Ravi,***

Nick Feamster

(Georgia Institute of Technology)

Alex Snoeren

(University of San Diego)

PERFORMANCE OF ONLINE SERVICES

Common online services: Search, shopping, online productivity:

- 100s of millions of users and growing
- Highly competitive environment: Content delivery matters!
- E.g: Google, Amazon, Facebook

Optimizing content delivery:

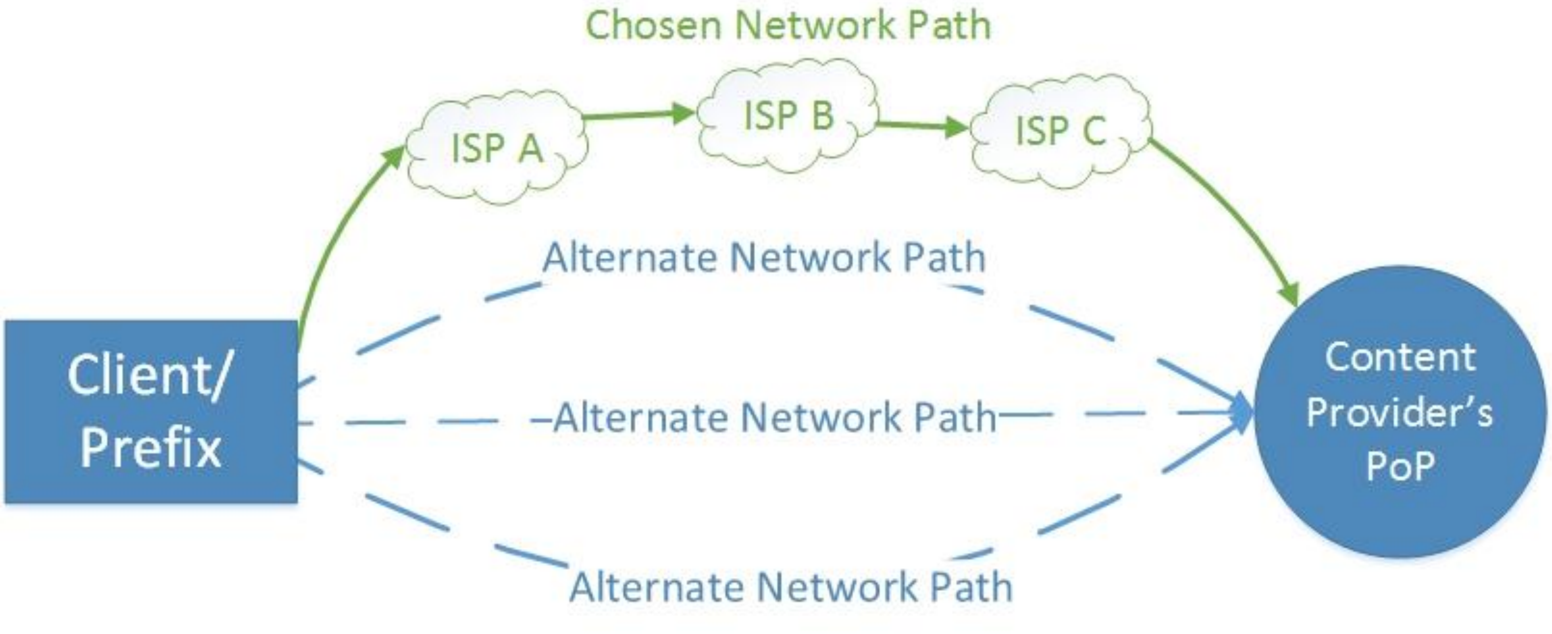
- Network routing for better connectivity (Traffic Engineering)
- Content routing to get closer to users

WHAT IS NETWORK ROUTING?

NANOG's bread and butter!

Evaluate and use better routes to content if available

BGP Traffic Engineering methods



BGP TRAFFIC ENGINEERING

Long term strategies:

- New peering connections
- New upstream connections
- New PoPs

Short term strategies:

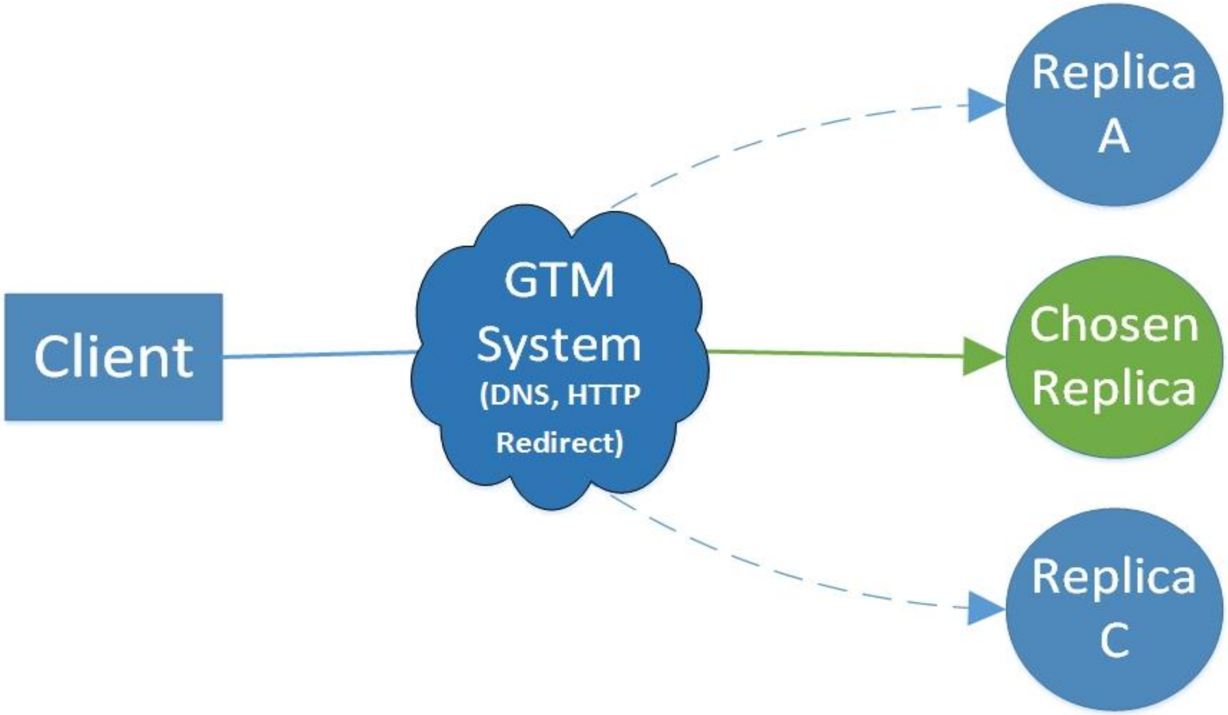
- BGP Local-pref tweaks
- Selective route announcement

WHAT IS CONTENT ROUTING?

Content available at multiple locations (Replicas)

Direct each client to its “ideal” replica

Global Traffic Management (GTM)



CONTENT ROUTING METHODS

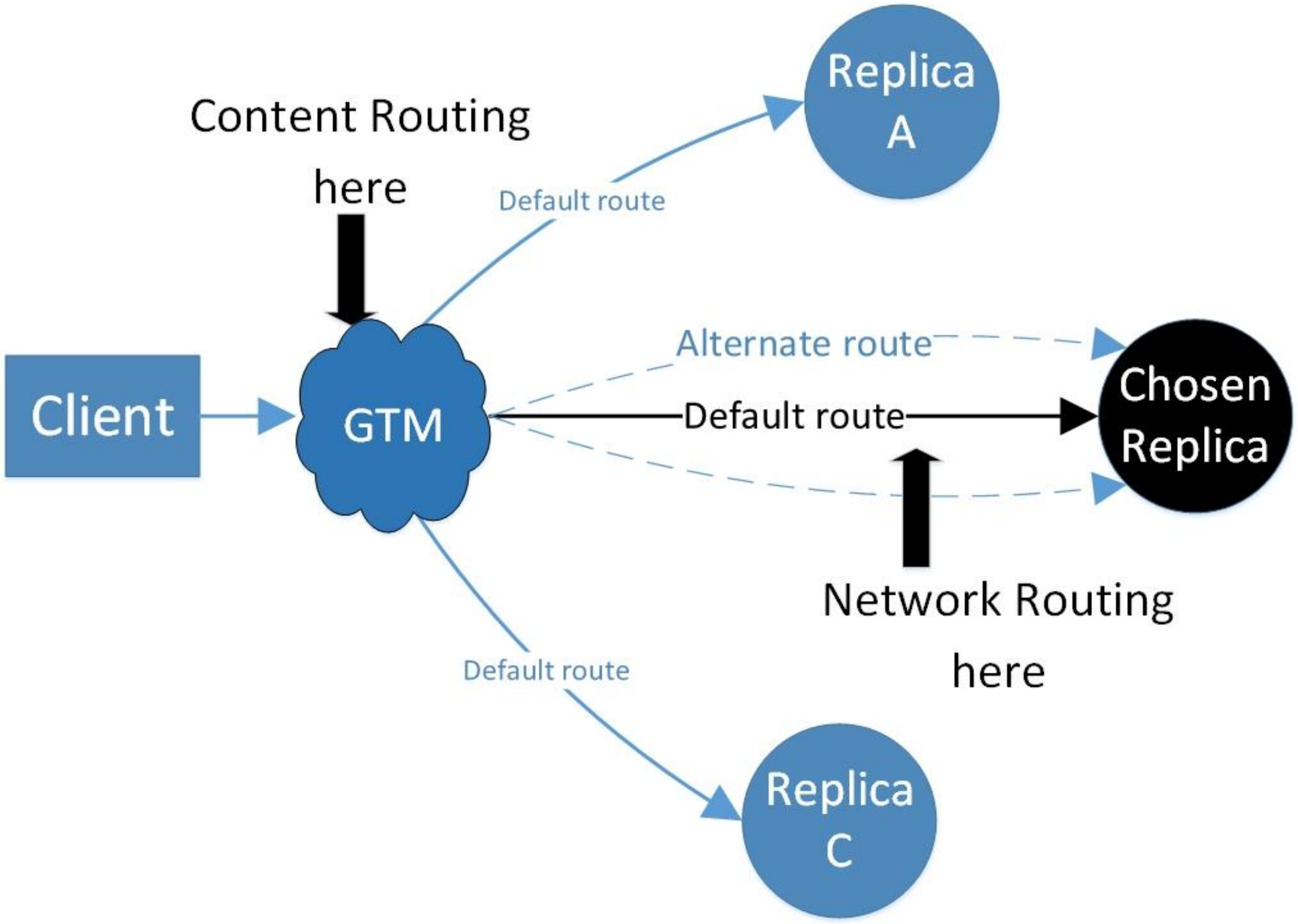
Redirect using:

- DNS, HTTP redirects/rewrites

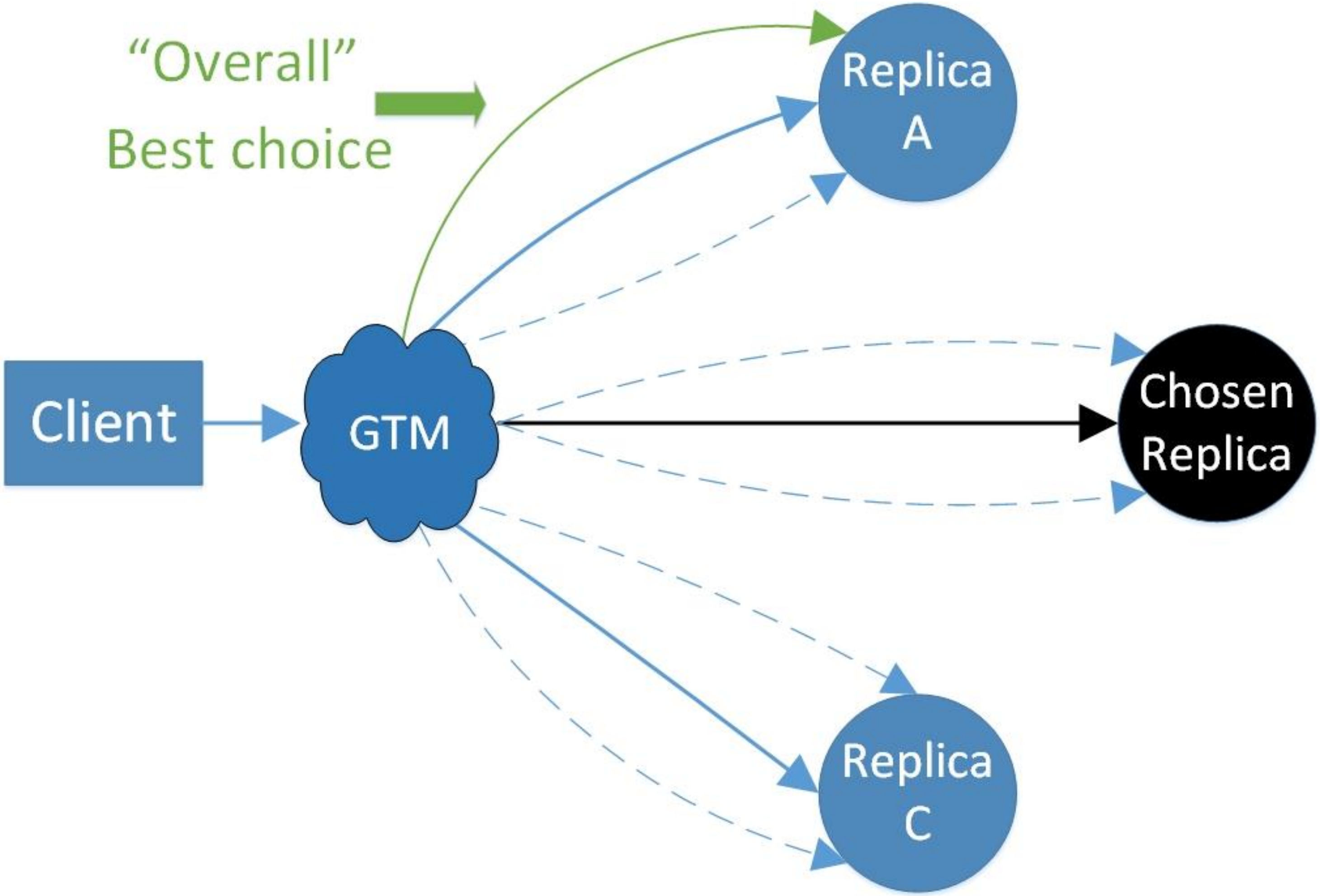
Redirect based on:

- Latency, Throughput, Load, Cost

A TYPICAL SCENARIO



THE PROBLEM


"Overall"
Best choice
Replica A
Client
GTM
Chosen Replica
Replica C

THE PROBLEM: CAN WE DO BETTER?

On one hand: Content Routing

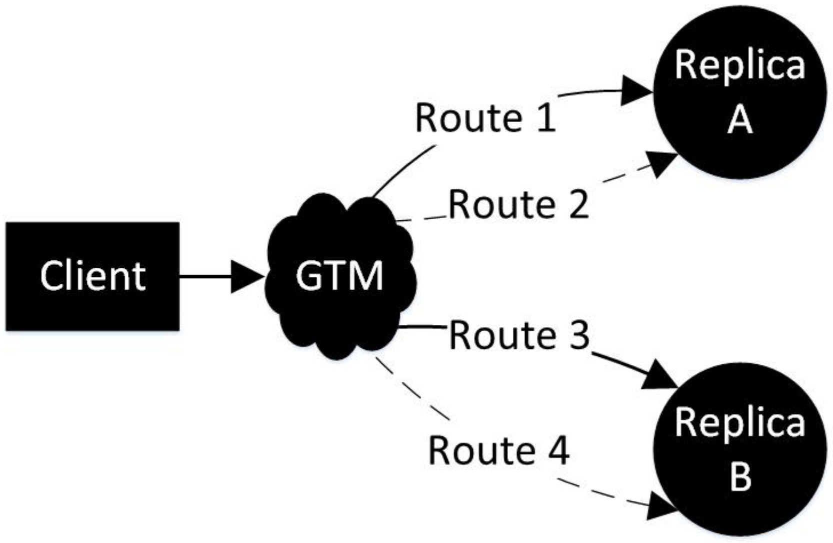
- GTM Systems have no visibility/control of network paths
- Measures performance against only current path

On the other hand: Network Routing

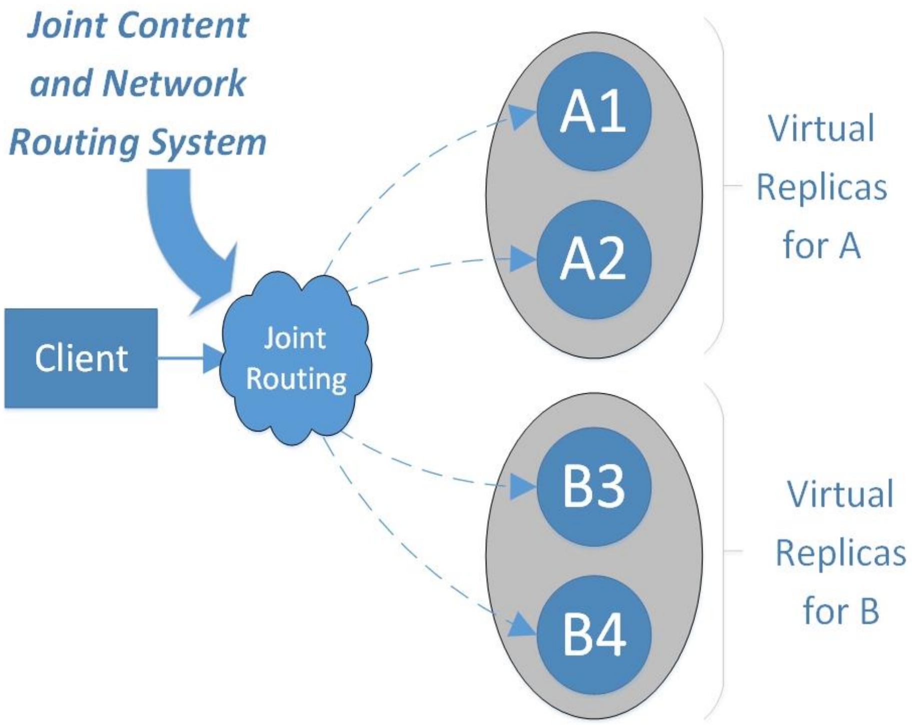
- Network operators have lot of tools to explore alternate paths
- But cannot see service performance at service/application level

Can we give TE capabilities of Network Routing to "higher level" GTM?

OUR SOLUTION: JOINT ROUTING



REALITY



OUR ABSTRACTION

OUR SOLUTION: USE VIRTUAL REPLICAS

1. Exhaustive search: Enumerate *all* network routes at *all* replicas
2. Evaluate performance of client prefix over these routes
3. Use existing GTM methods to select appropriate Virtual Replica (Replica + Route combination)

"Joint" because above steps are performed by a single system

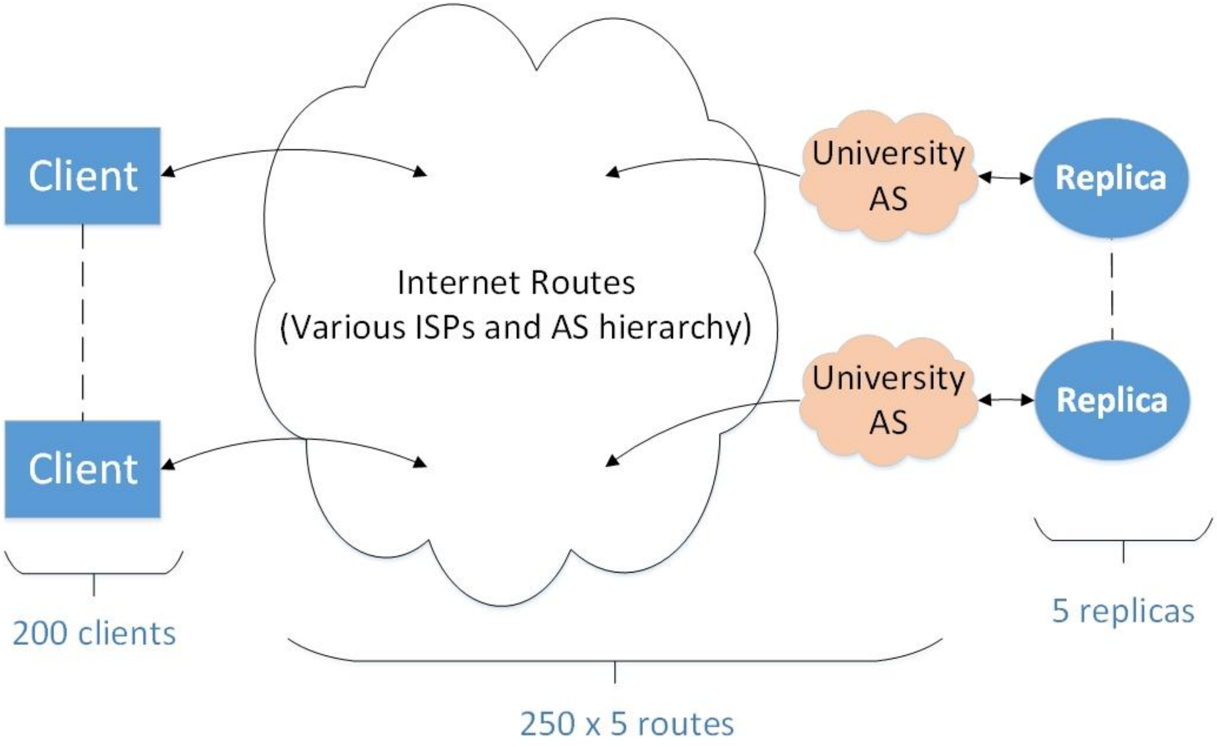
OUR GOAL

What gains does Joint Routing offer?

Establish a Replicated service testbed and measure

THE SETUP

1. Five replicas: 3x East Coast, 1x Mid-West, 1x West Coast
2. 200 clients: “PlanetLab” nodes around the world
3. At each replica, evaluate around 250 routes using clients.



THE SETUP: EXPLORING ROUTES

How do we explore these 250 routes?

Available techniques:

Egress route selection:

- Local-pref
- Weights
- Tunneled egress …

Ingress route selection:

- Selective prefix announcements
- AS PATH Prepending
- BGP Community attributes …

THE SETUP: EXPLORING ROUTES

How do we explore these 250 routes?

Available techniques:

Egress route selection:

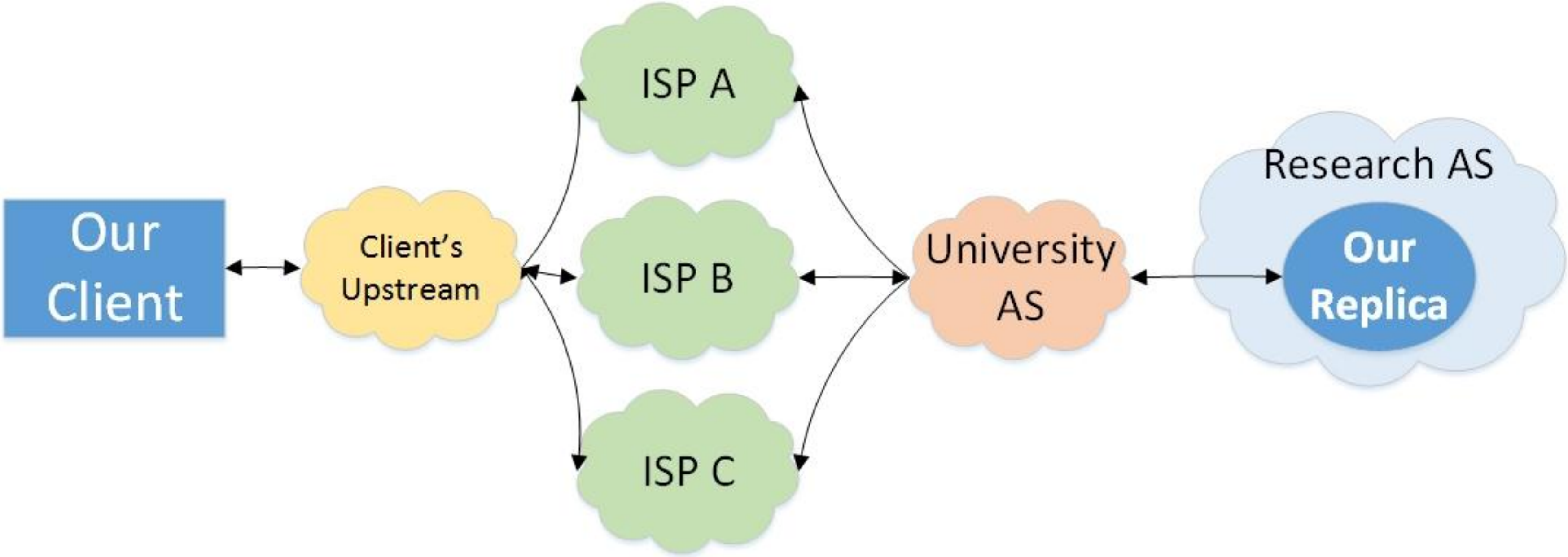
- Local-pref
- Weights
- Tunneled egress …

Ingress route selection:

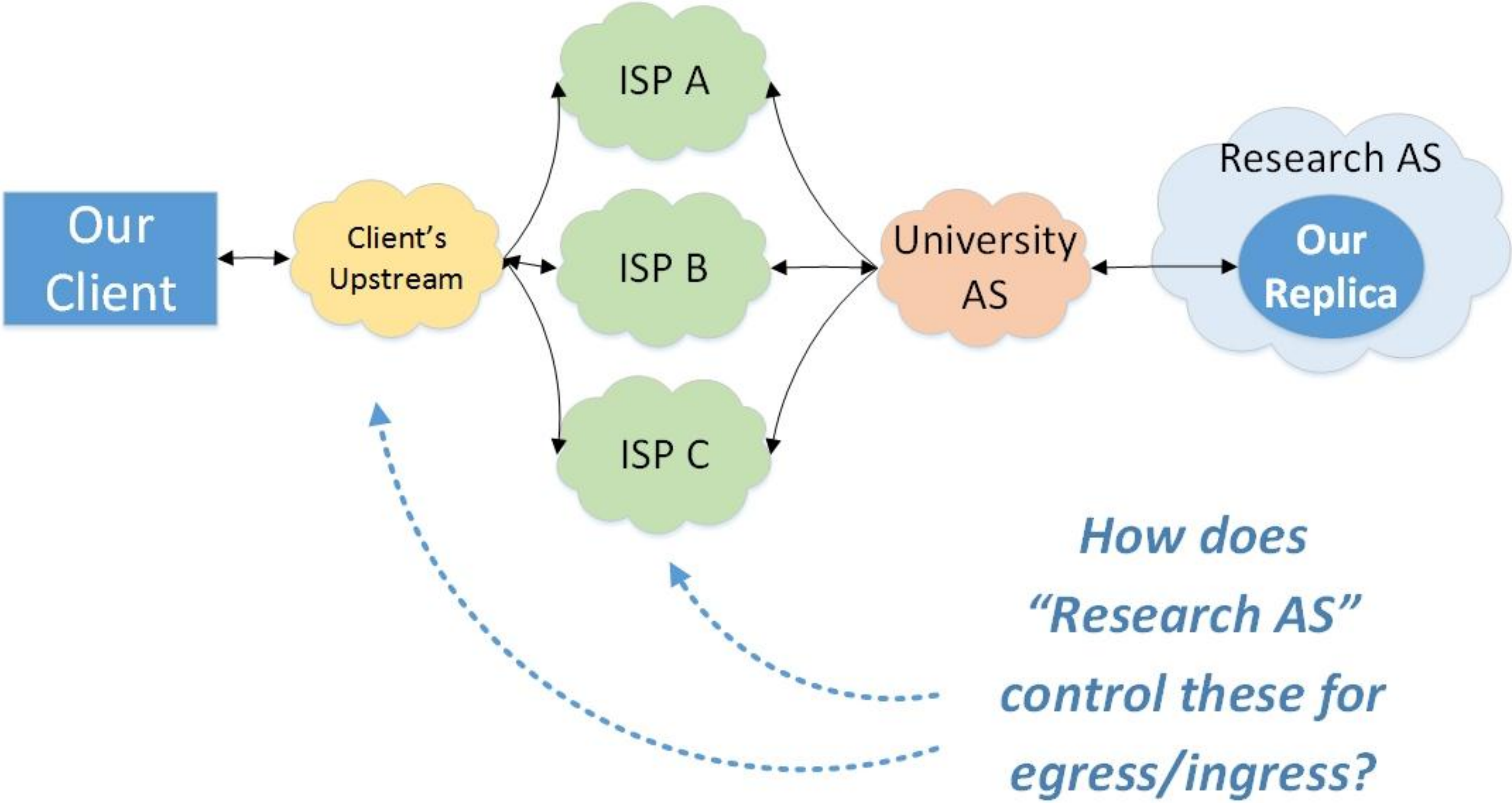
- Selective prefix announcements
- AS PATH Prepending
- BGP Community attributes …

Our Setup does not permit using these!

THE SETUP: EXPLORING ROUTES



THE SETUP: EXPLORING ROUTES



How does "Research AS" control these for egress/ingress?

THE SETUP: EXPLORING ROUTES

Research AS does not peer directly:

Local-Pref, Community, Weights, etc are ignored

THE SETUP: EXPLORING ROUTES

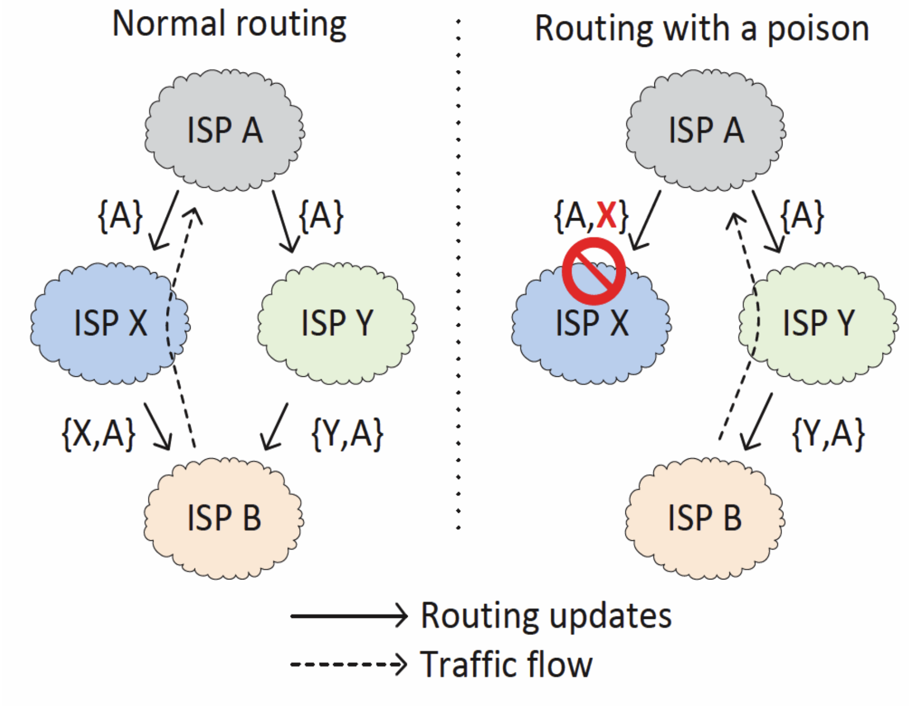
Research AS does not peer directly:

Local-Pref, Community, Weights, etc are ignored

Use BGP AS Path Poisoning, focus on ingress routes

THE EXPERIMENT: AS-PATH POISONING

Poison an AS to force route around it



THE EXPERIMENT: MEASUREMENTS

Evaluate every client performance for each Virtual Replica

Evaluate Latency (ping), Throughput & Jitter (iperf). Traceroute for topology.

At each replica:

- 1.5 million pings
- 0.5 million iperfs
- Over a period of 3 months.

THE RESULTS

Start with single “best” replica: *RTT is 107.3ms (avg)*

THE RESULTS

Start with single “best” replica: *RTT is 107.3ms (avg)*

With Network Routing at this replica: *4.3% reduction*

THE RESULTS

Start with single “best” replica: *RTT is 107.3ms (avg)*

With Network Routing at this replica: *4.3% reduction*

With Content Routing with 5 replicas: *16.7% reduction*

THE RESULTS

Start with single “best” replica: *RTT is 107.3ms (avg)*

With Network Routing at this replica: *4.3% reduction*

With Content Routing with 5 replicas: *16.7% reduction*

Now add Joint routing: ***20.4% reduction***

Joint routing yields a 3.7% point RTT improvement over content routing

THE RESULTS

Increase in throughput:

- Baseline (avg): 212.4 Mbps
- Network routing: +0.8%
- Content routing: +8.1%
- Joint Routing: +11.2%

Jitter reduction:

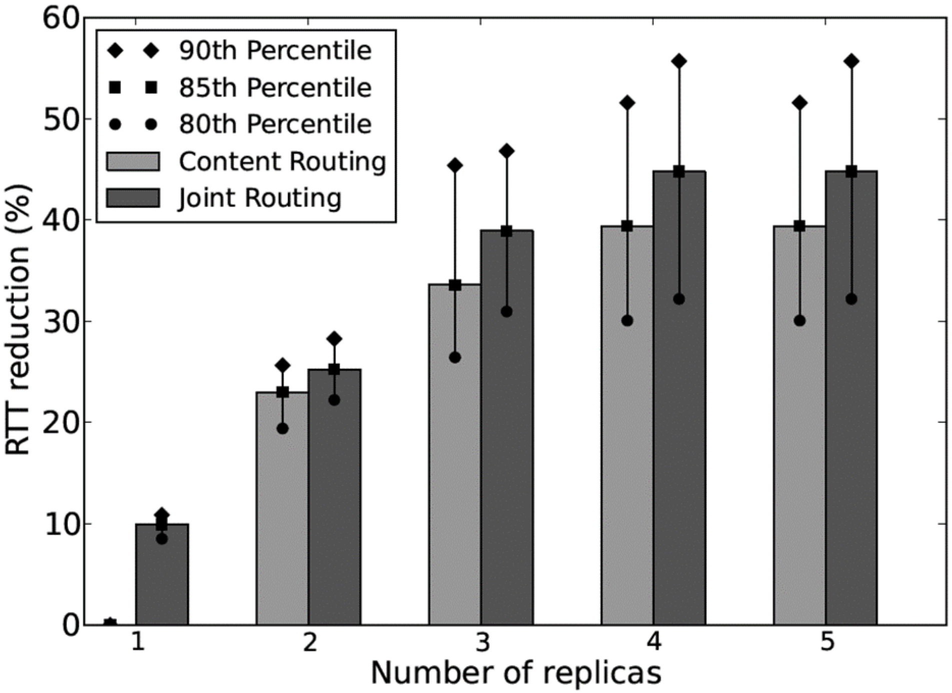
- Baseline (avg): 5.9ms
- Network Routing: -9.3%
- Content routing: -11.8%
- Joint routing: -17.5%

THE RESULTS : MARGINAL GAINS

Joint routing gives marginal improvement over content routing

As we increase number of replicas, margin does not disappear

7% of clients moved to a different replica



THE RESULTS : LIMITED ROUTE ANNOUNCEMENTS

How much “chaos” does this cause?

- 200 clients x 5 replicas x 250 routes is a lot!
- How many Poisons do we need to announce to?

Limited route announcements:

- Five poisons (at each replica) = 60% of maximum improvement possible
- With 7-8, almost full 3.7% gain

THE RESULTS : SUMMARY

Joint routing yields:

- 3.7% pts RTT reduction
- 11.29% pts Throughput increase
- 17.57% pts Jitter reduction

compared to Content routing, as Marginal gains

5 Poisoned announcements yield 60% of maximum improvement

WHAT NEXT? QUESTIONS TO ANSWER

How to make Joint routing practical?

- Have content providers already attempted/considered this?
- AS PATH poisoning is not the best way to explore routes. What are alternatives methods/test-beds we could use?

When is Joint routing useful?

- Our testbed sees a 3.7% RTT reduction. What use cases find this useful?
- What do improvements look like in a real-world setting?

QUESTIONS/FEEDBACK?

Contact Information:

Bharath Ravi (bravi@gatech.edu)

Vytautas Valancius (vytautas.valancius@gmail.com)

Nick Feamster (feamster@cc.gatech.edu)

Alex Snoeren (snoeren@cs.ucsd.edu)