Scalable Simulation Of Complex Network Routing Policies

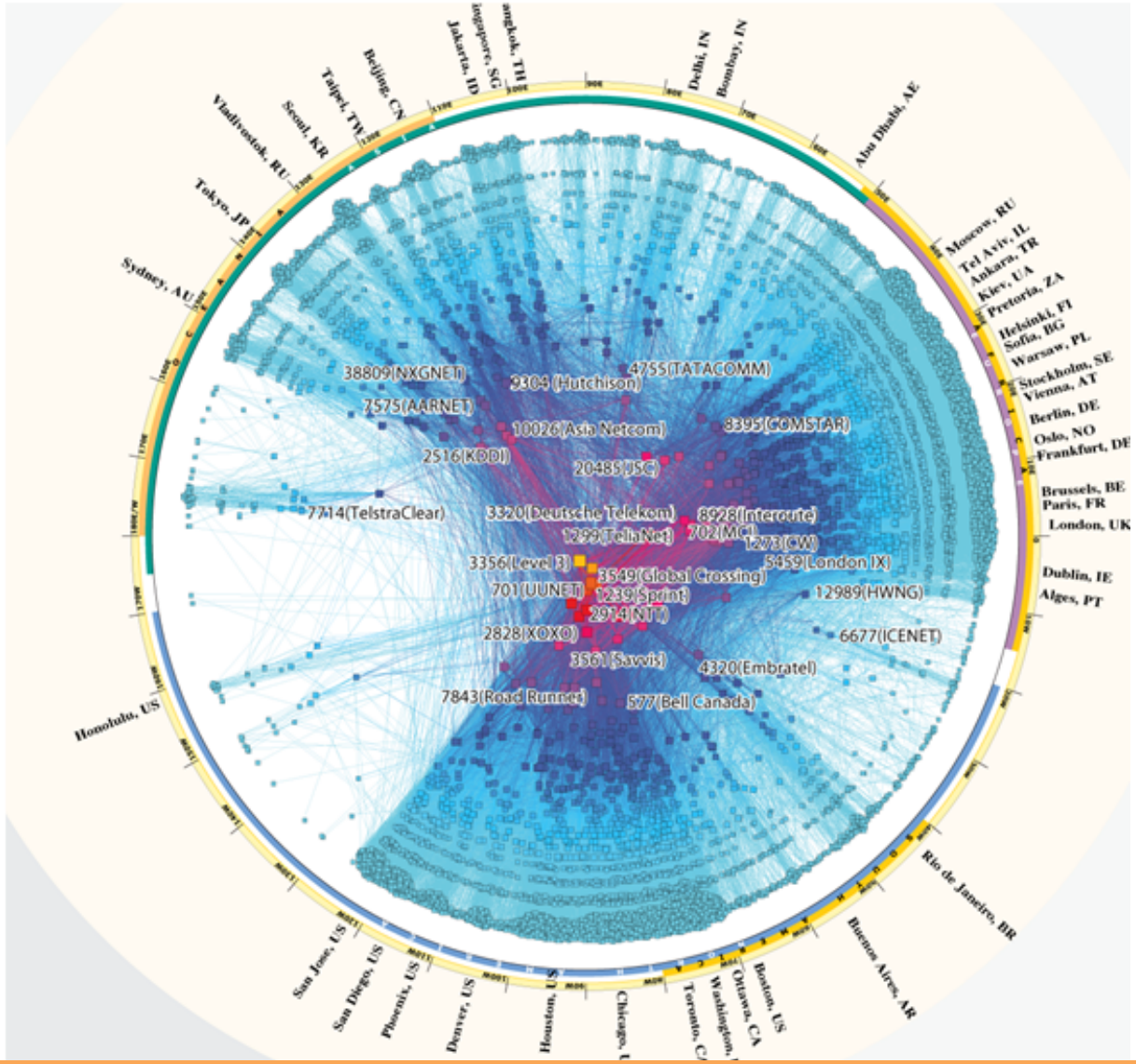
Andrew Stone, Michelle Strout, Steven DiBenedetto, and Daniel Massey

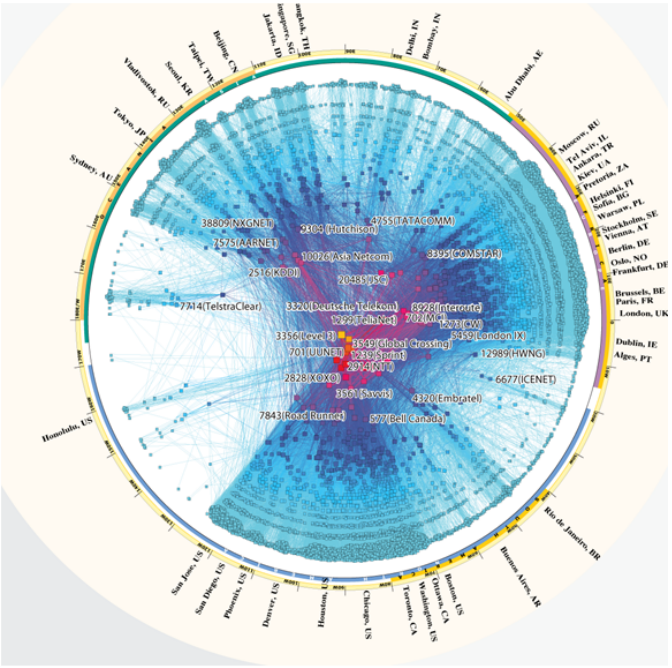
Colorado State University

May 19, 2010

Computing Frontiers

People who study networks, want to analyze this:

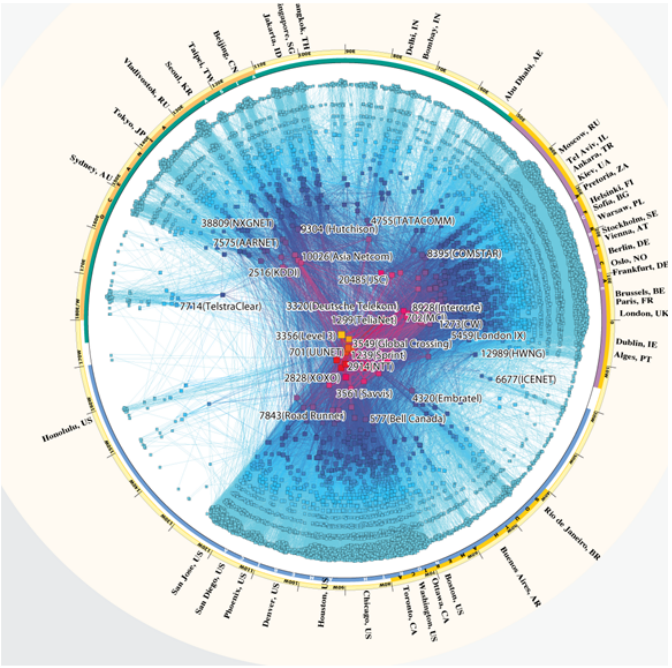




About this model:

- This is a model of the internet
- Experimentally produced
- Nodes are autonomous systems (ISPs)
- 33K nodes

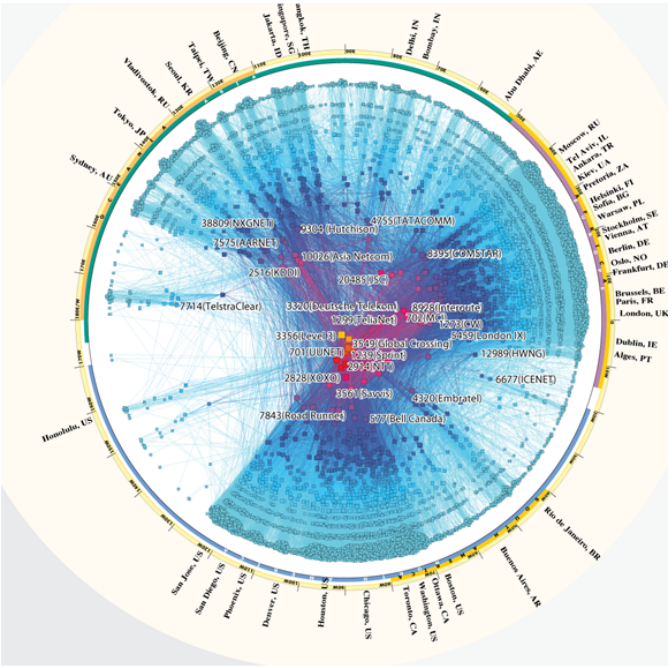
source: http://www.caida.org/home/



Analyze what?

Researchers want to know:

- How **accurate** this modeled topology is
- how **topology** affects routing tables
- how **policy** affects routing tables



Ideally, they want a simulator that:

- shows resulting routing tables
- can work with large topologies (30k - 100k nodes)
- can work with various policies
- can produce results quickly (overnight)
- works with accessible hardware

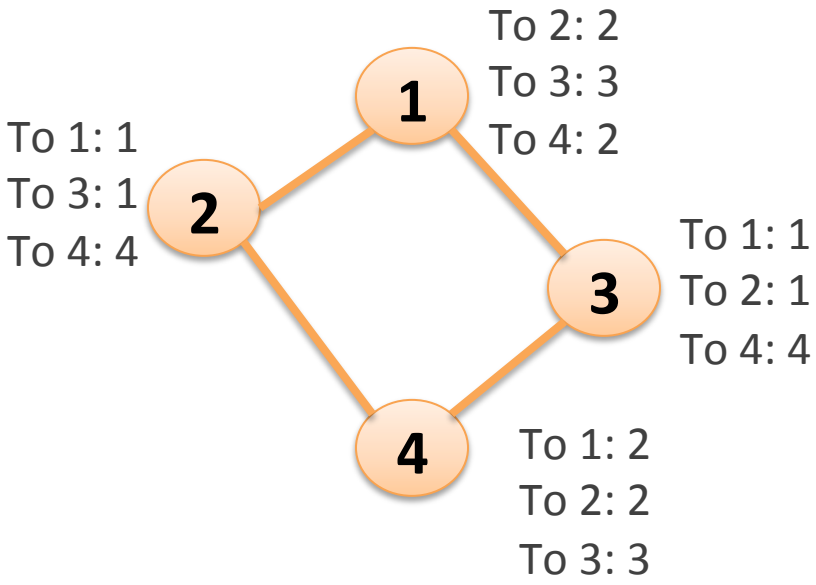
The Problem

Existing simulators:

- Only work with small topologies
- Have hardcoded policies

Complicating factors:

- How to specify policy?
- Policies are more than just shortest path
- O(n^2) memory requirements for routing tables



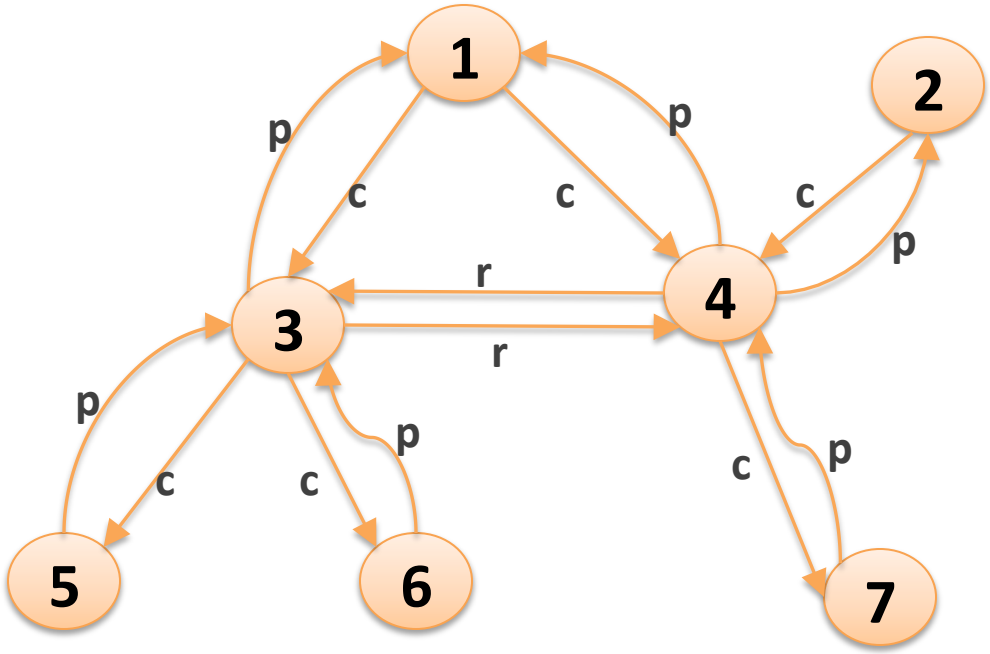
MR. Sim

Our tool: MR.Sim

- **Inputs topologies**: specification file
- **Inputs policies**: Metarouting based framework
- **Parallelism**: embarrassingly parallel through **simulation segmentation**
  - reduces O(n^2) memory requirements to O(n) at any one time

How a topology is modeled

Nodes = autonomous systems
Edges = connections
Edge labels describe the connection and influence policy



Examples of connections

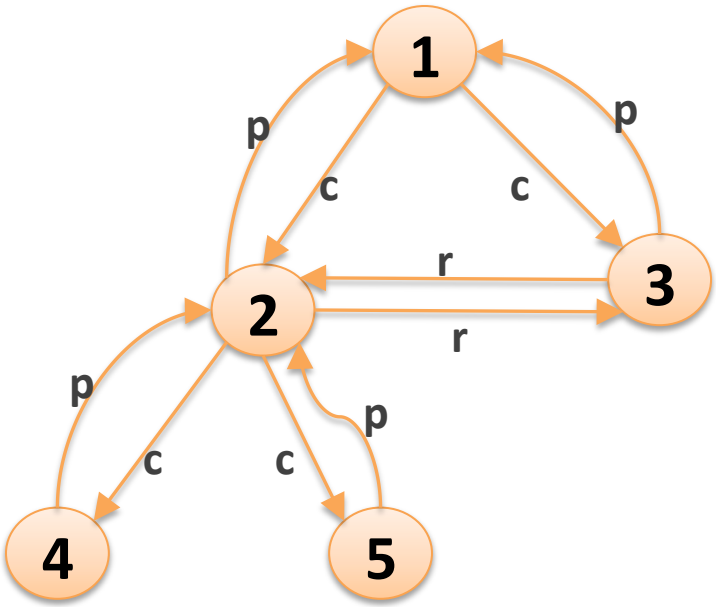
c: customer connection

r: peer connection

p: provider connection

How topologies are input to Mr. Sim

Map files list the edges in the network graph and the policies (labels) that are on those edges.



SimpleNet.map

```
1 2 c
2 1 p
1 3 c
3 1 p
2 3 r
3 2 r
2 4 c
4 2 p
2 5 c
5 2 p
```

Routing Policies

RAML: Routing Algebra Meta Language

Routing algebras

Composition operators

Routing algebras represent policies

RAML policies are guaranteed to converge

Routing Algebras

$$(\Sigma, \preceq, L, \oplus, \mathcal{O})$$

All about **signatures** and **labels**

Labels describe: connections

Signatures describe: routes

Signatures give routes some value

A route is: (destination, next-hop, signature)

Routing Algebras

$$(\Sigma, \preceq, L, \oplus, \mathcal{O})$$

Σ Set of possible signatures

$\preceq$ Preference relation (partial ordering of sigs)

L Set of possible labels

$\oplus$ Label operator

$\mathcal{O}$ Originator set (possible signatures initial routes can have)

Propagating routes

A route is: (destination, next-hop, signature)

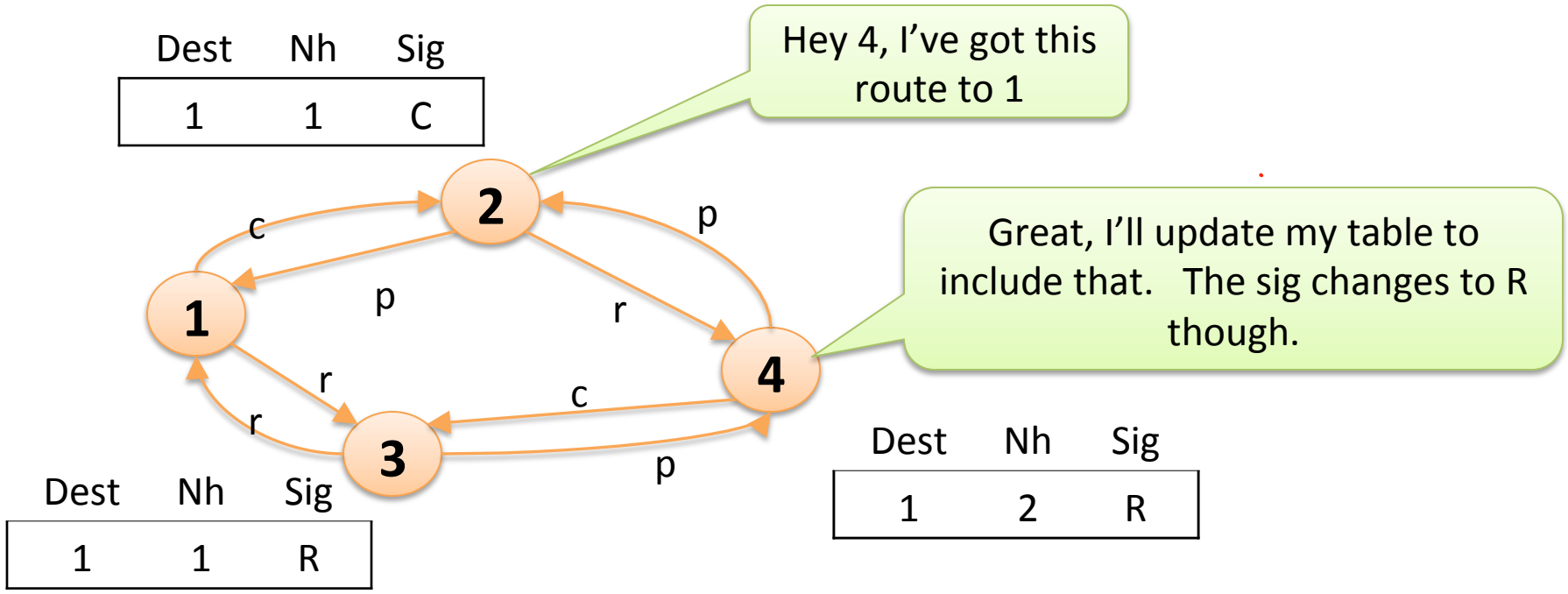
Σ = {C, R, P} L = {c, r, p}

Signature determines how preferred a route is

$\preceq$ C < R < P

smaller is more preferred.

Nodes propagate routes to neighbors. Neighbors update their routing tables based on the propagated information.



Propagating routes

A route is: (destination, next-hop, signature)

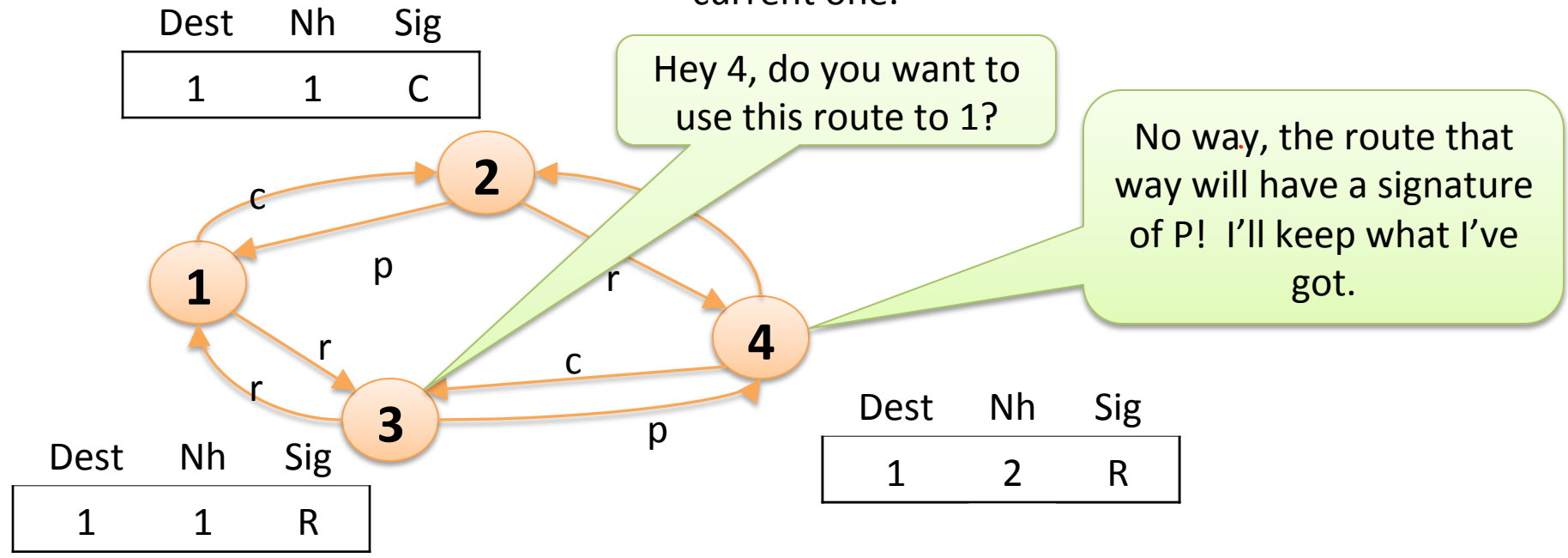
Σ = {C, R, P} L = {c, r, p}

Signature determines how preferred a route is

$\preceq$ C < R < P

smaller is more preferred.

When a route is propagated it is compared against any existing routes to the same destination. The table is only updated if the new route is more preferred than the current one.



Mr. Sim Classes:

Mr. Sim model's routing algebras in a C++ framework. Users override these classes to define various properties of the algebra.

The framework includes the following abstract classes:

Signature
Label
PreferenceRelation
LabelOperator
NetworkReader

Algebra

Algebra is passed a PreferenceRelation, LabelOperator, and NetworkReader

BGP Policy

FM(LP(3)) Σ = {0, 1, 2} L = {0, 1, ...}

$\oplus$ (lbl, sig) = rhs > lhs then phi else lhs

SIMSEQ(0, N-1) Σ = {0, 1, .., N-1} L = {0, 1, ..., N-1}

$\oplus$ (lbl, sig) = addtolist(lbl)

RouteID() Σ = {0, 1, ..} L = {0, 1, ...}

$\oplus$ (lbl, sig) = lbl

BGP = FM(LP(3)) X SIMSEQ(0, N-1) X RouteID

BGP Policy

```
1 FMIntOp fmOp;
2 SimSeqOp seqOp;
3 LpOp lpOp;
4 LPInt prefRel;
5 SeqPref seqPref;
6 BGPRelationshipReader relationshipReader;
7 RouteIDReader routeIDReader;
8
9 // create the three algebras BGP is composed of
10 Algebra algFMLP(
"FMLP", &fmOp, &prefRel, &relationshipReader);
11 Algebra algAdd(
"SIMSEQ", &seqOp, &seqPref, &routeIdReader);
12 Algebra algRouteID(
"RouteID", &lpOp, &prefRel, &routeIDReader);
13
14 // calculate the composed routing algebra (the BGP
15 // policy). First parameter specifies number of
16 // algebras to compose
17 Algebra bgp = Algebra::lexProduct(
3, &algFMLP, &algAdd, &algRouteID);
```

Simulation Segmentation

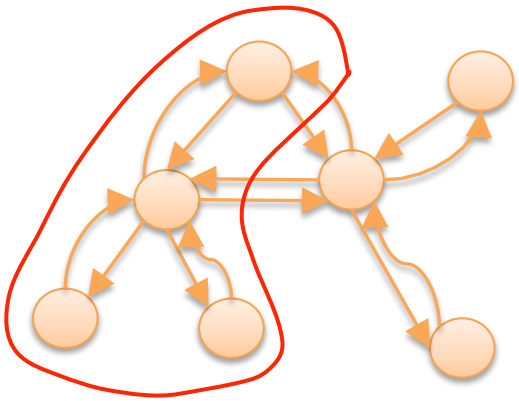
network	memory usage
1k nodes	57 MB
10k nodes	6.88 GB
30k nodes	could not test
internet (33k nodes)	could not test

All route's simulation has O(n^2) memory requirement

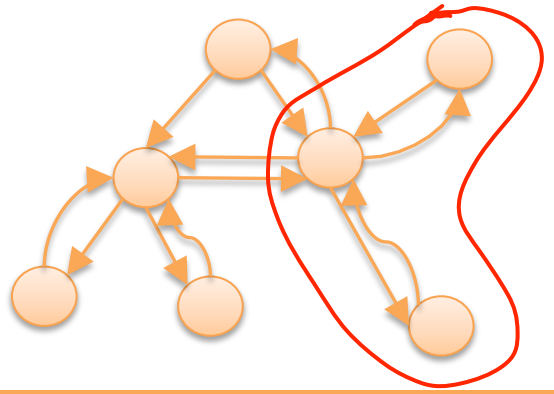
Swapping kills performance

Due to how we model the routing problem it is possible to do a set of simulations that each only solve the problem for a set of routes.

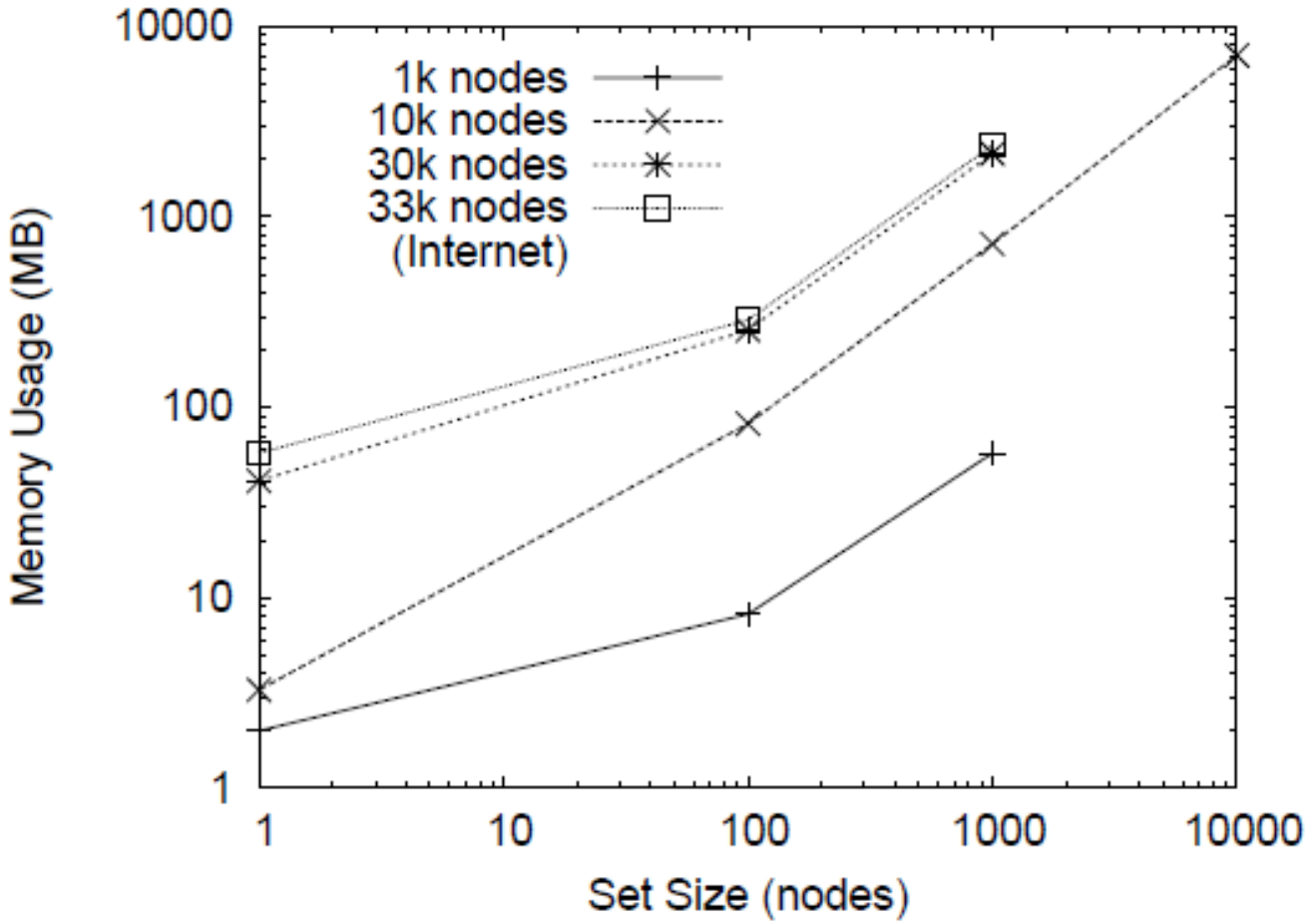
First just find routes to these nodes:



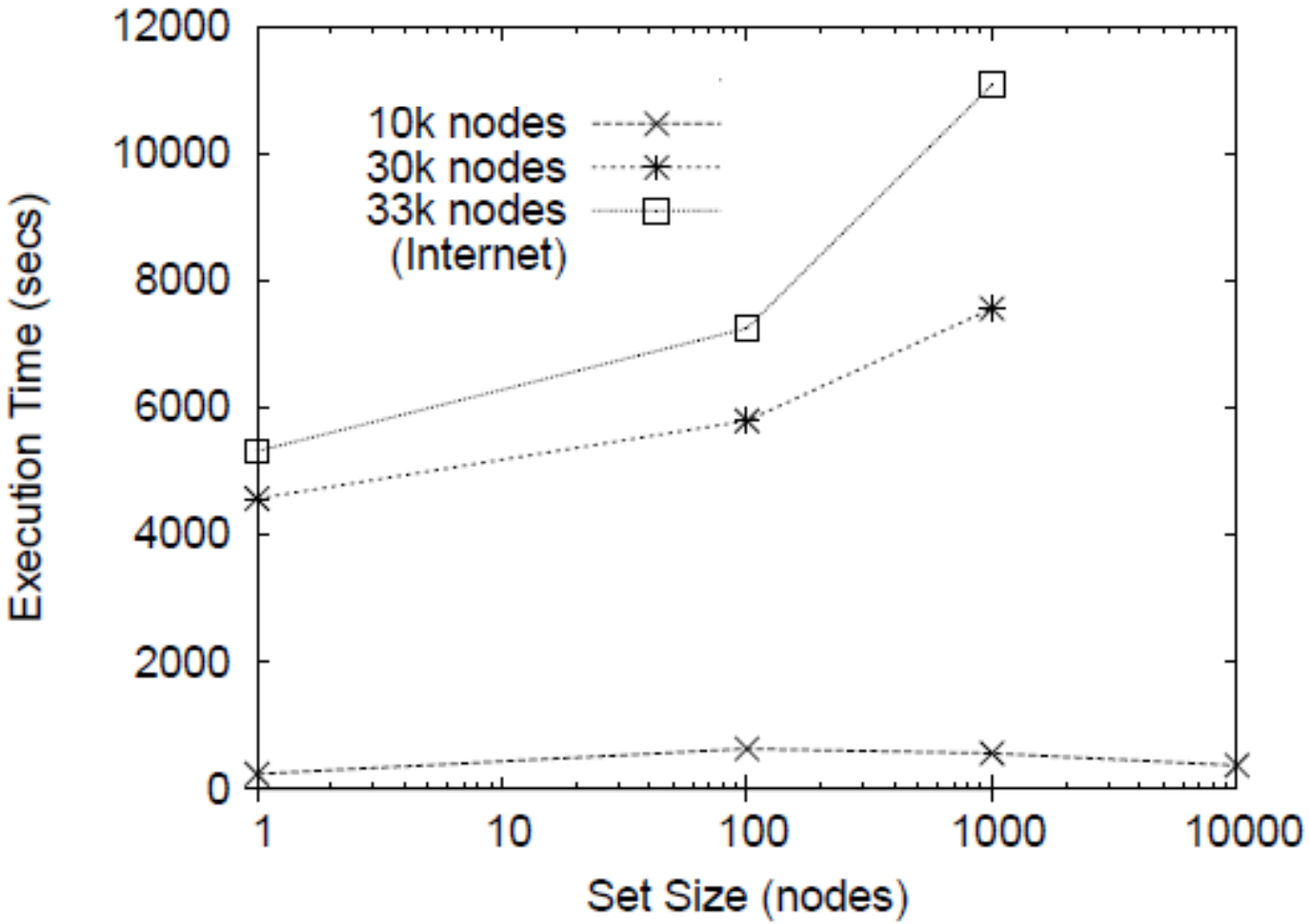
Then find routes to these nodes:



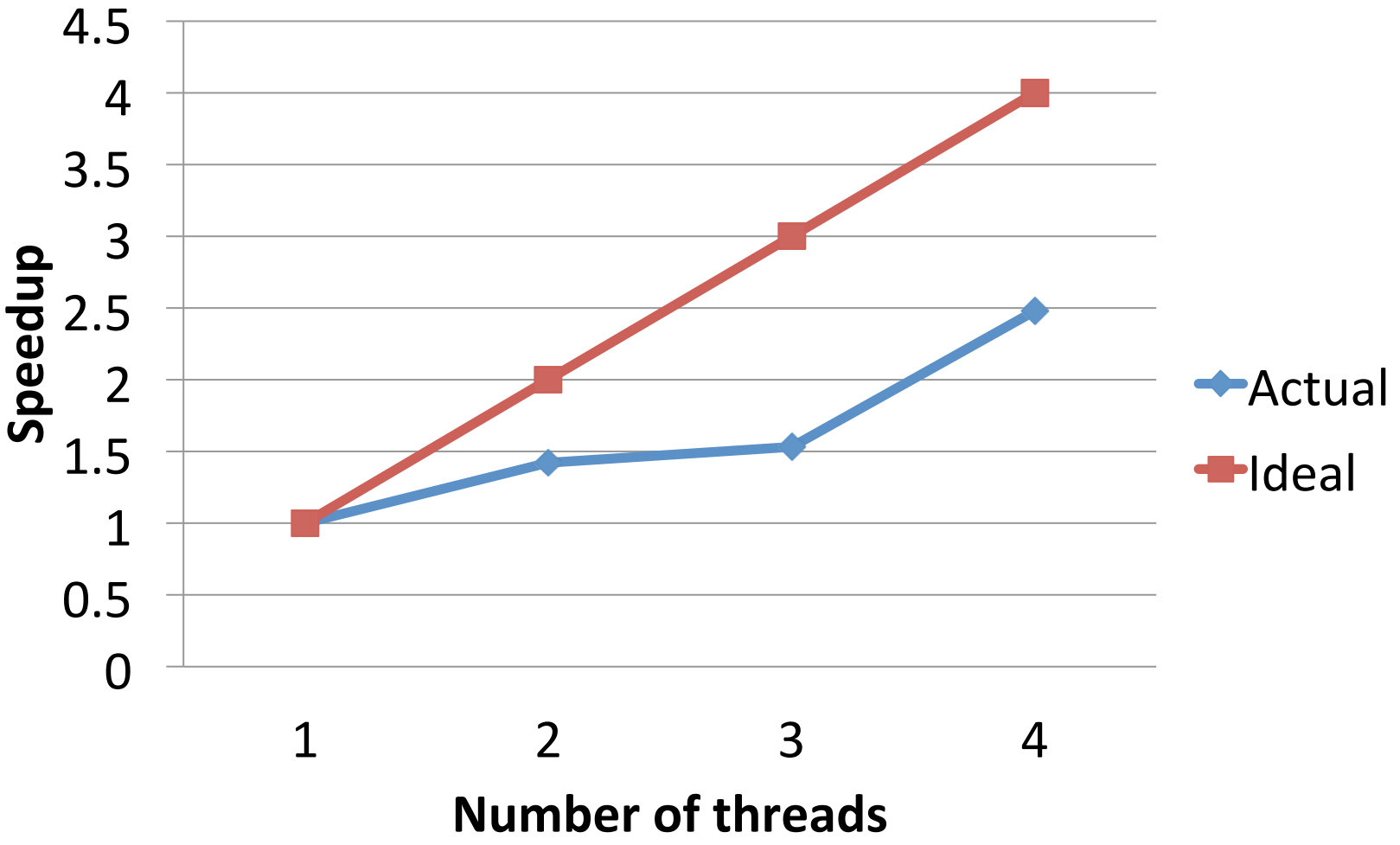
How set size affects memory usage (simple policy)



How set size affects performance (simple policy)



Shared Memory Scalability

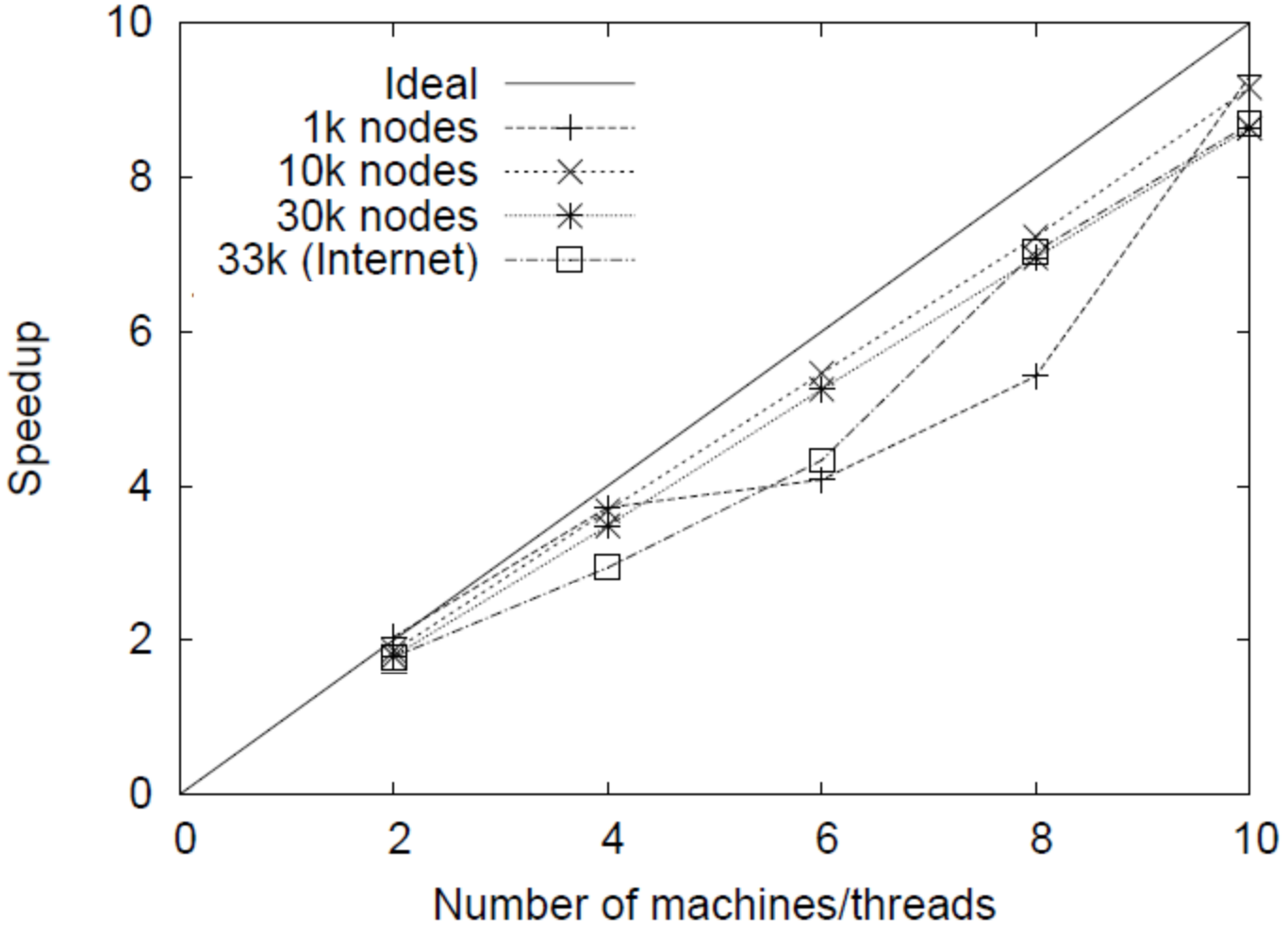


Shared Memory Scalability

			L2 data cache misses (billions)		
Threads	Execution time	Speedup	total	per thread	serial total / per thread
1	3307.26	1	29.49	29.49	1
2	2324.89	1.42	37.4	18.7	1.58
3	2156.54	1.53	52.21	17.4	1.69
4	1334.85	2.48	41.21	10.3	2.86

Effective cache size is preventing ideal speedup
(cache per thread goes down as # of threads goes up)

Distributed Memory Scalability



Conclusions

Our tool:

- Can input topologies via specification file
- Includes a C++ framework based on metarouting
- Can operate with BGP on internet topology
- This tool is being used to conduct internet-routing research
- Topologies are only going to get more complicated

Scalability:

We didn't get it on a shared memory architecture
(due to shared cache)

Do did get it on a distributed memory architecture

We've succeeded in our goal (43 min simulation on 10 machines)

Questions?