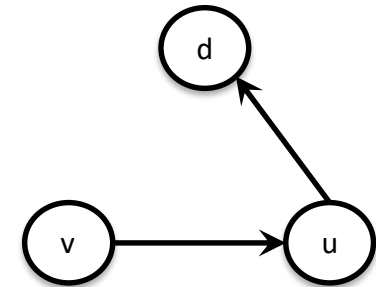
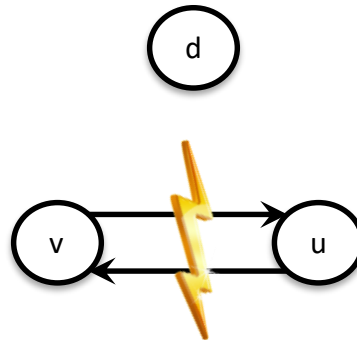
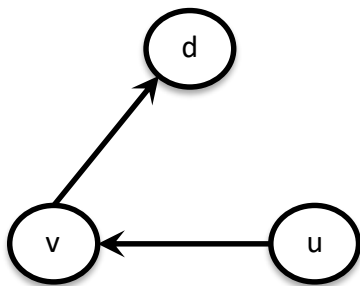


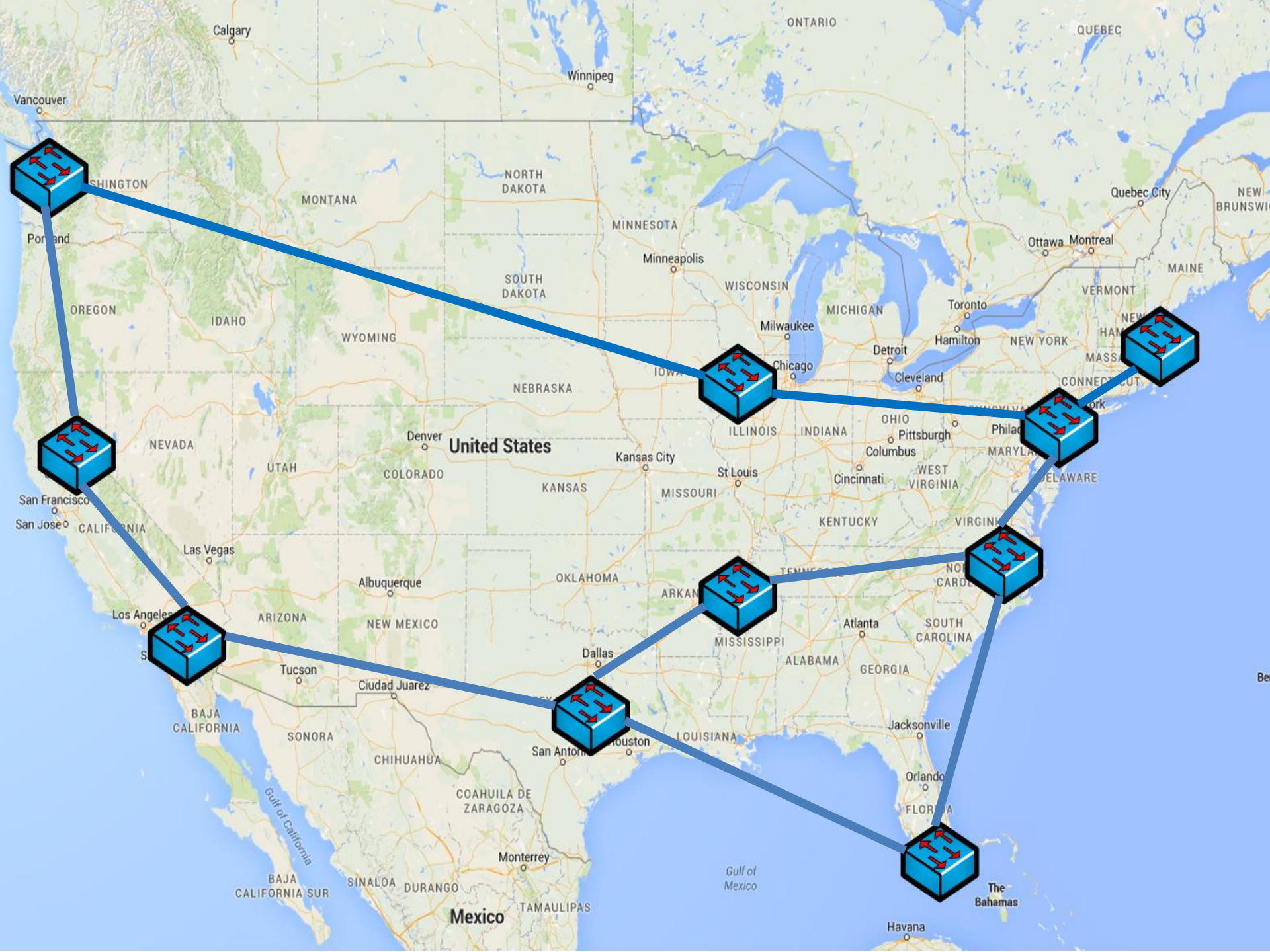
Consistent Updates in Software Defined Networks: On Dependencies, Loop Freedom, and Blackholes

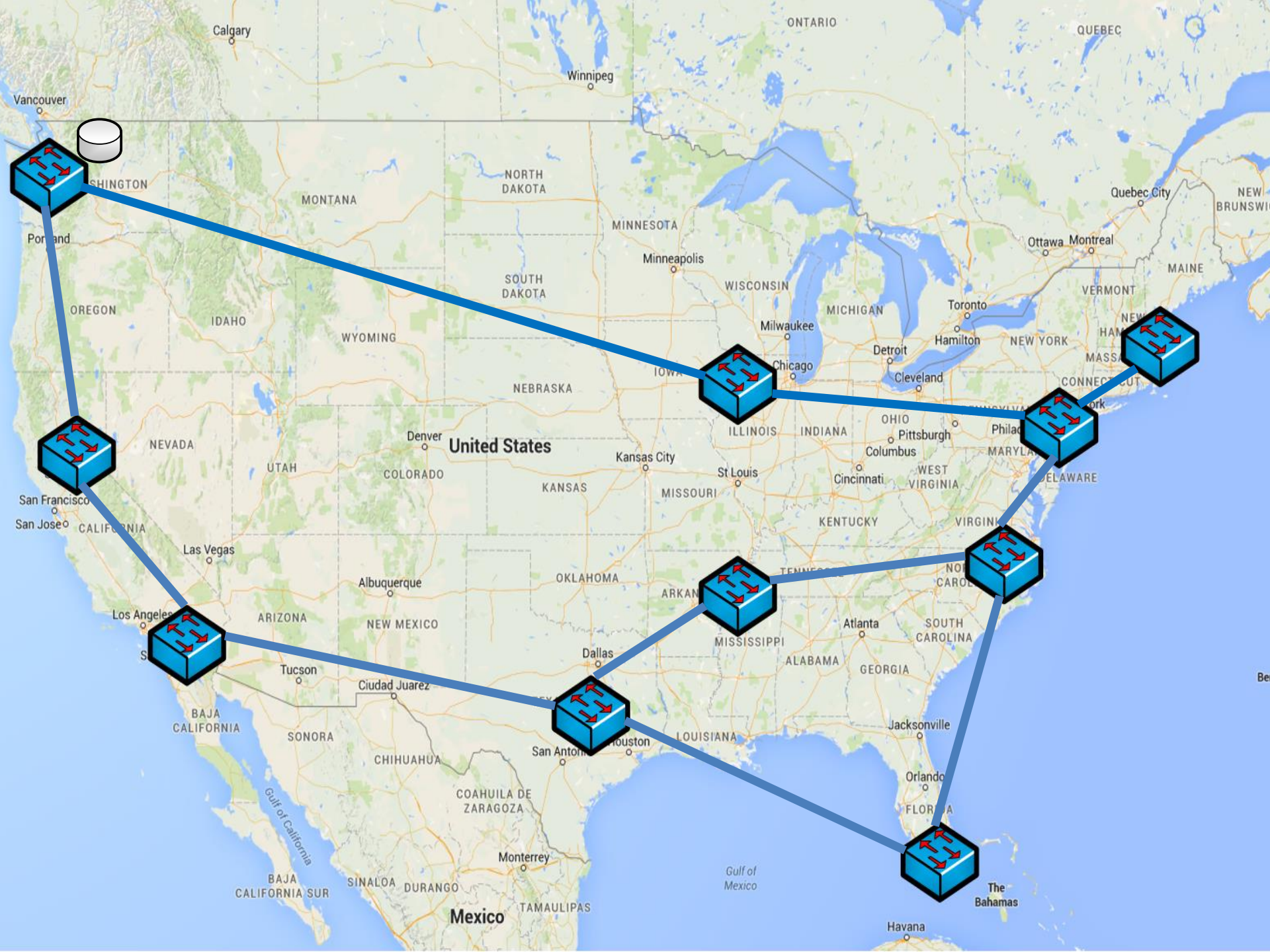


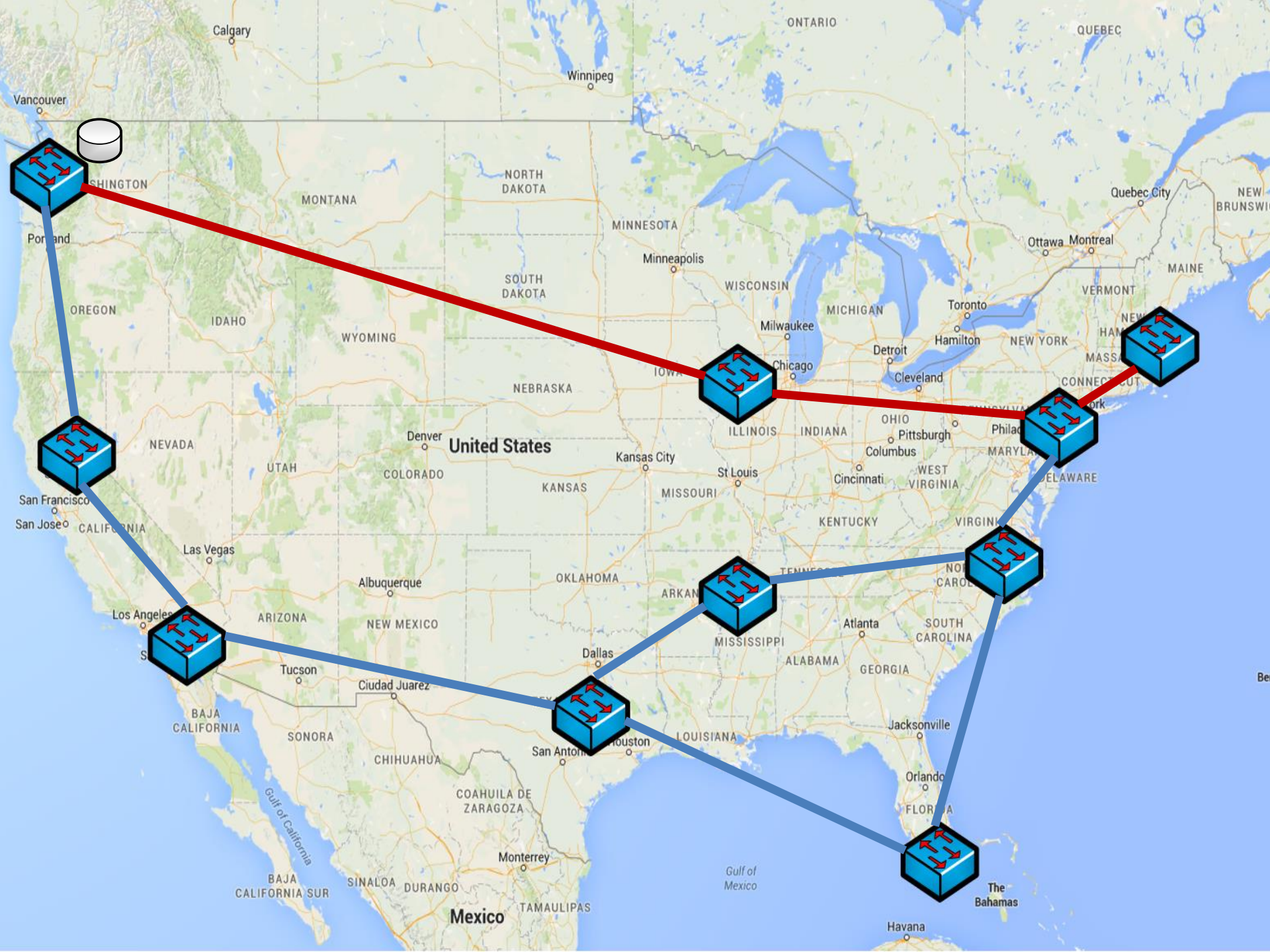
Klaus-Tycho Förster, Ratul Mahajan, and Roger Wattenhofer

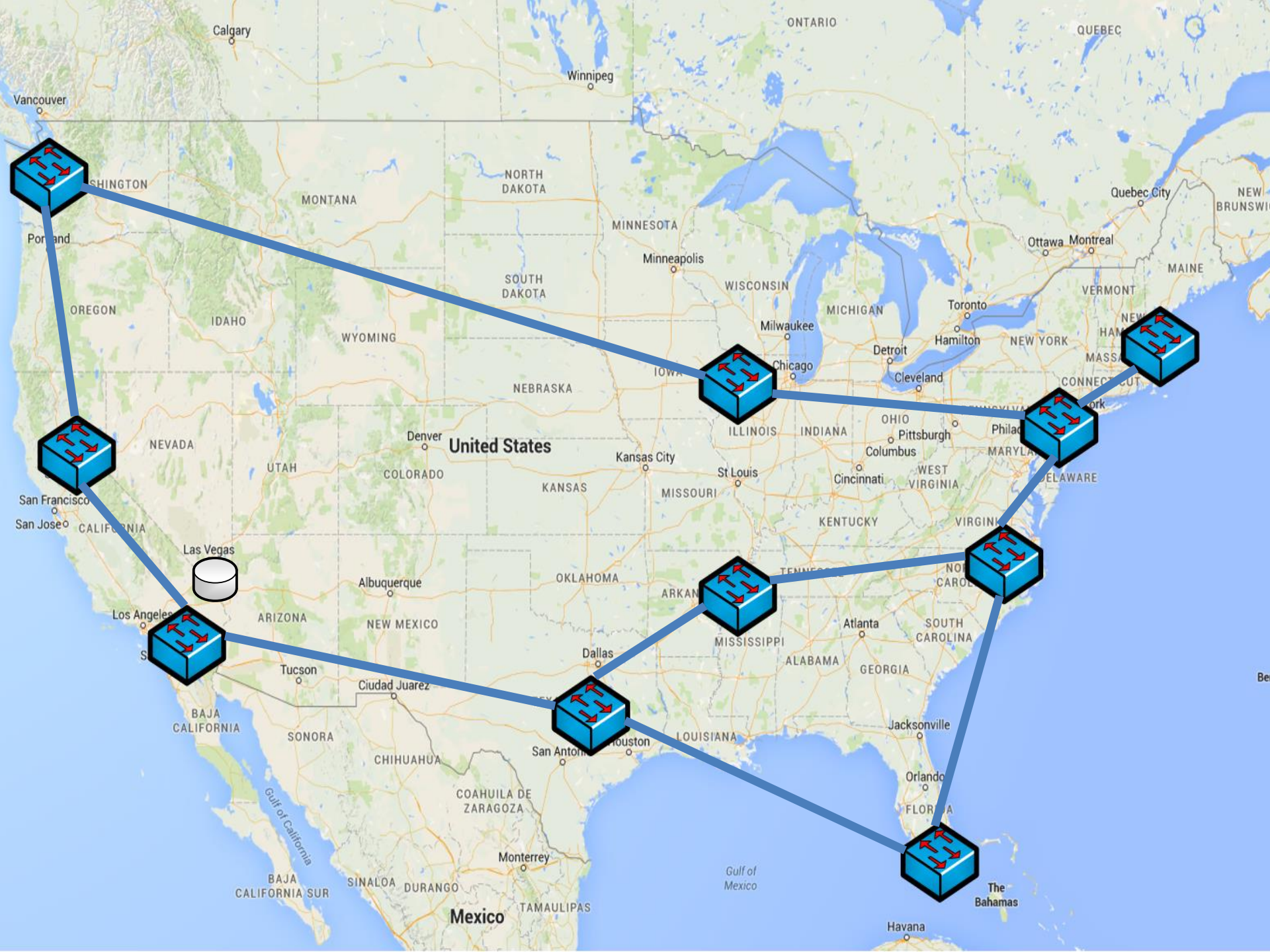
Consistent Updates in Software Defined Networks: On Dependencies, Loop Freedom, and Blackholes

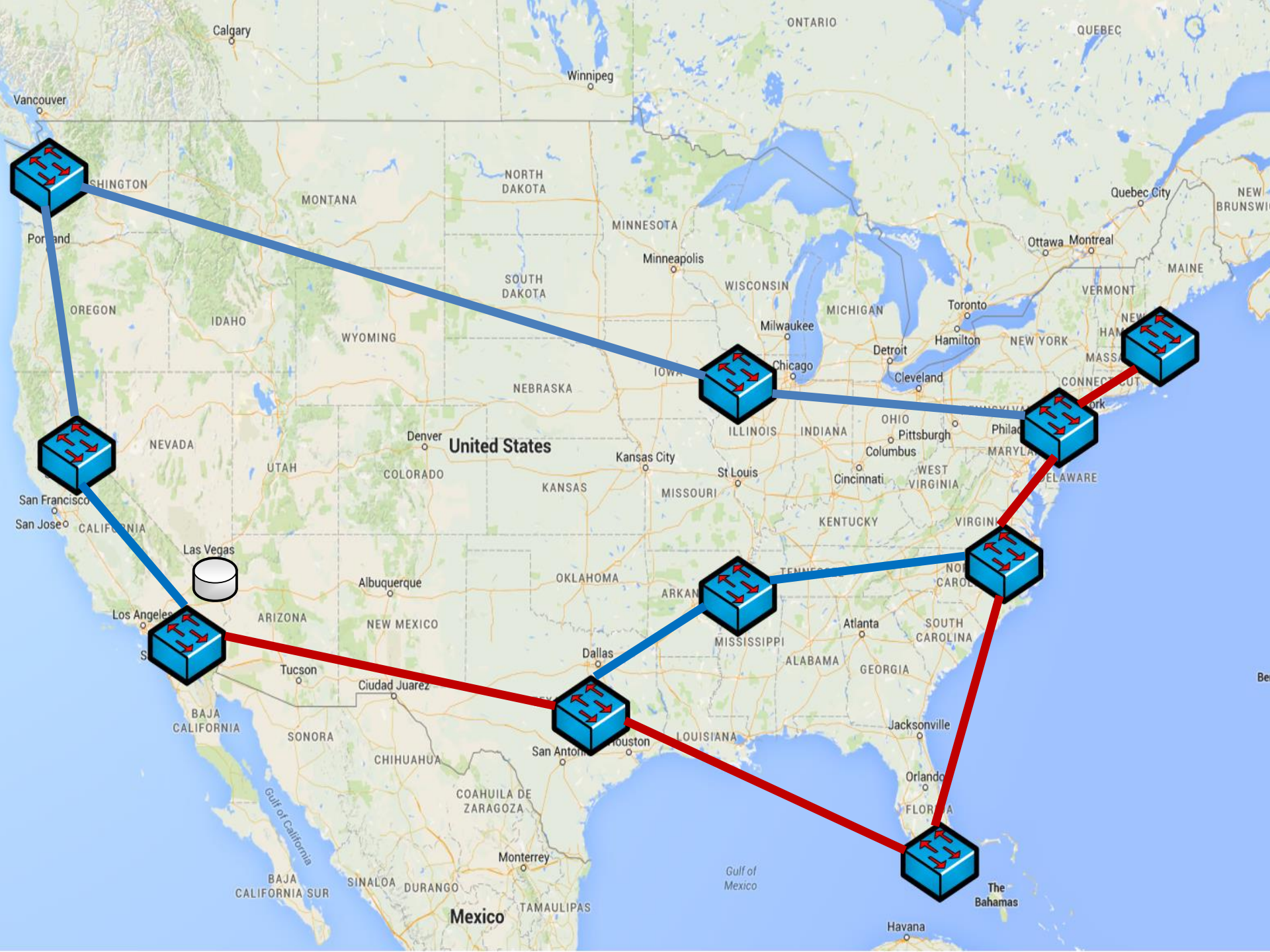


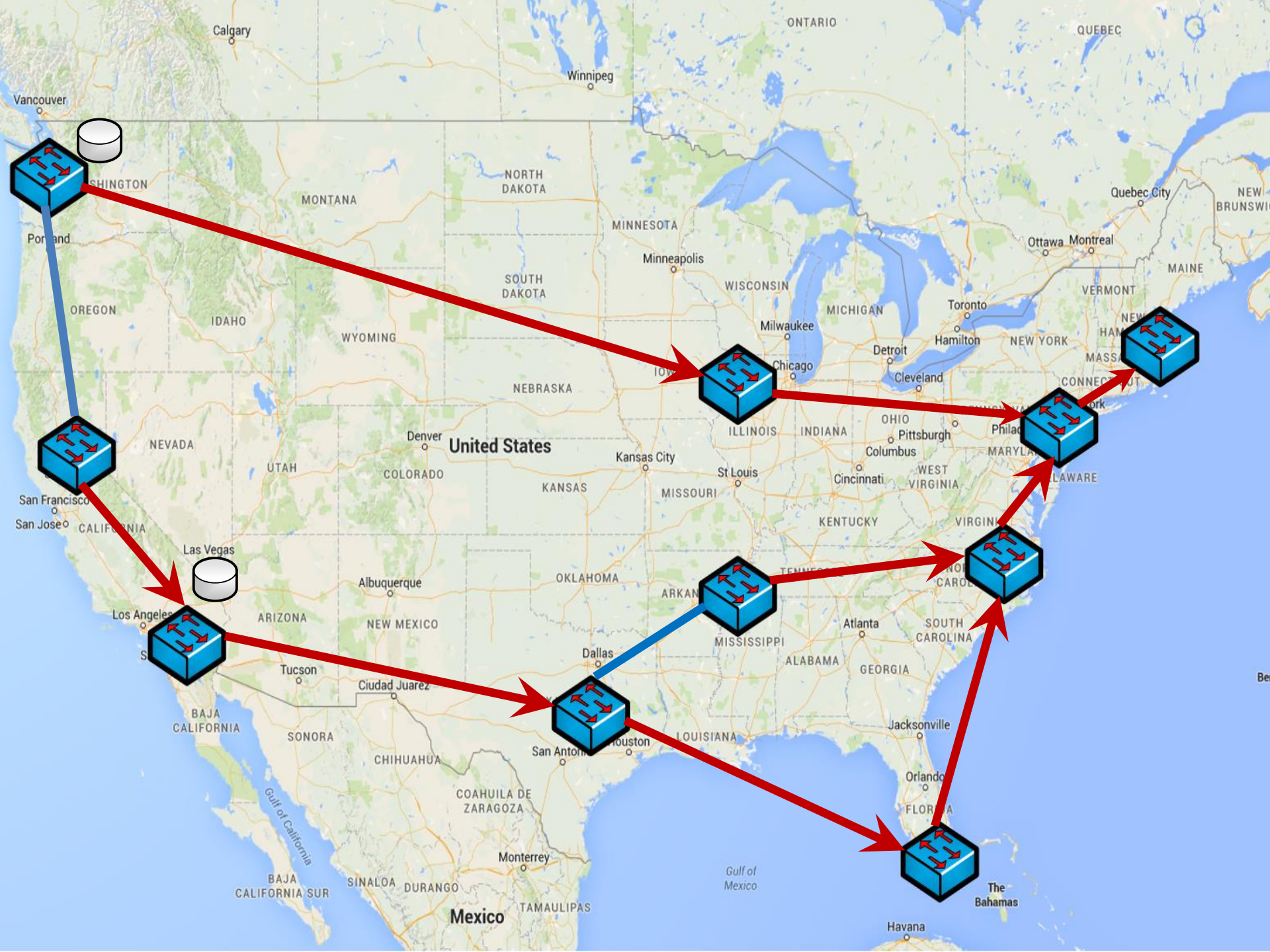


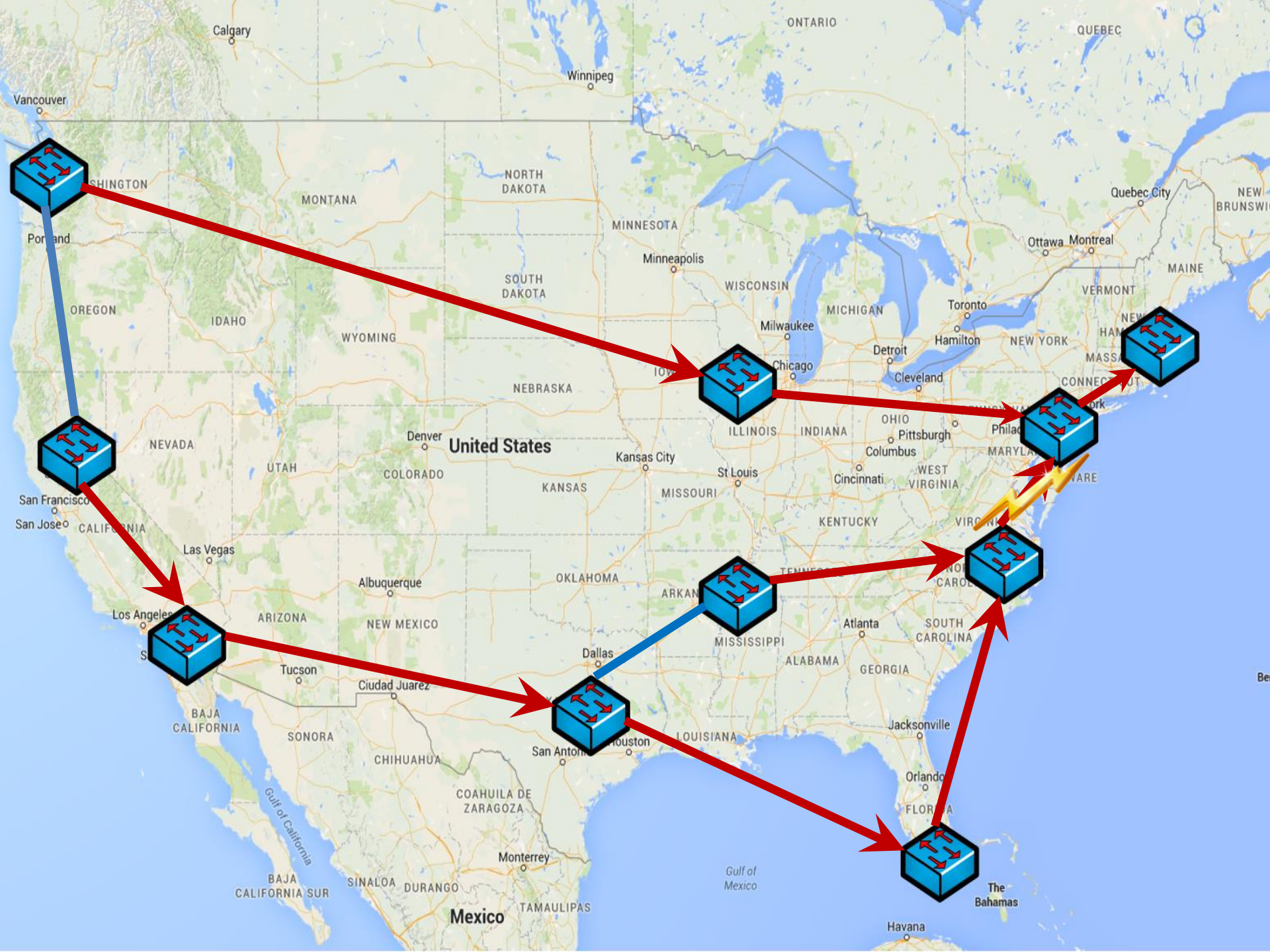


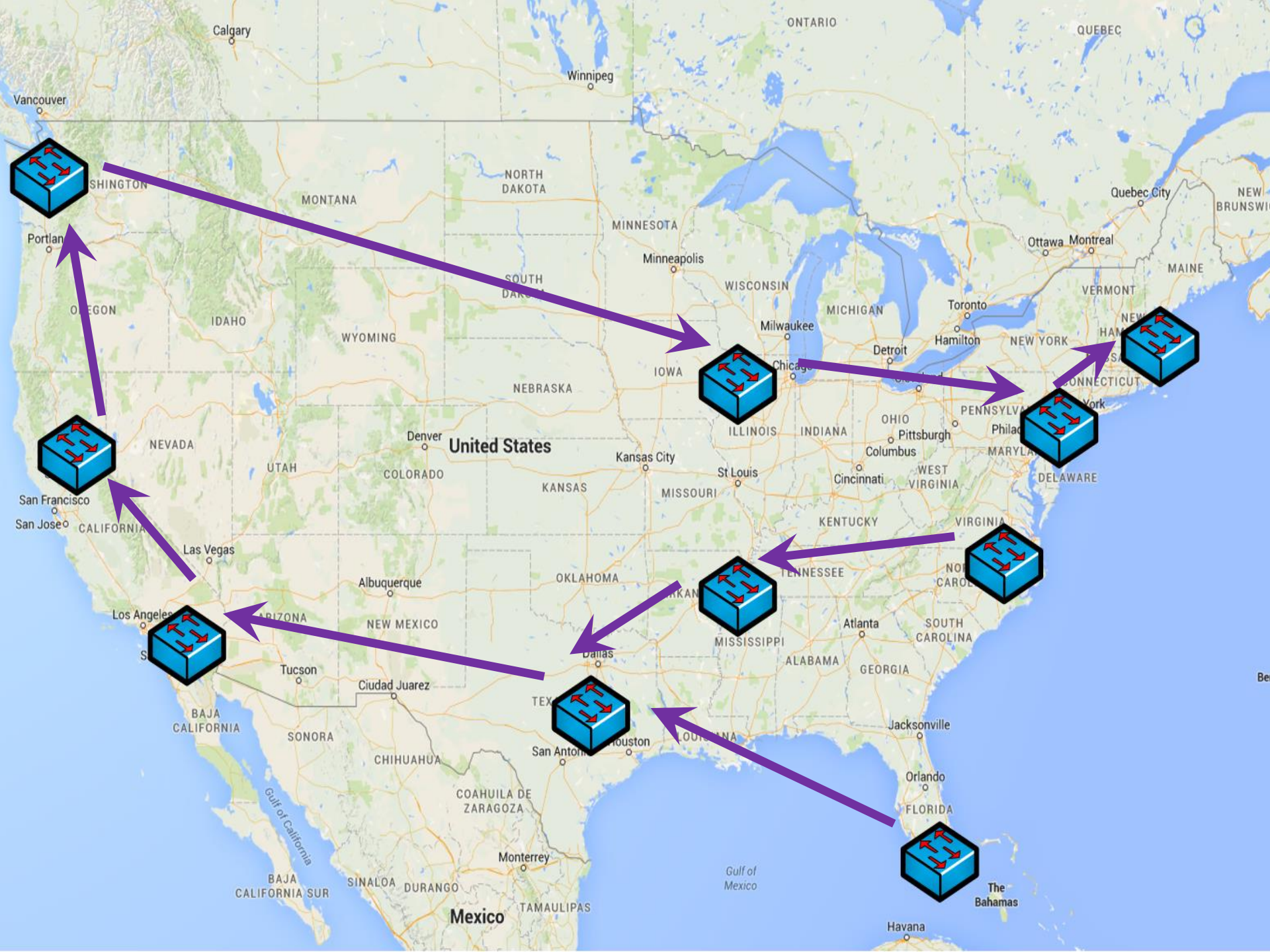








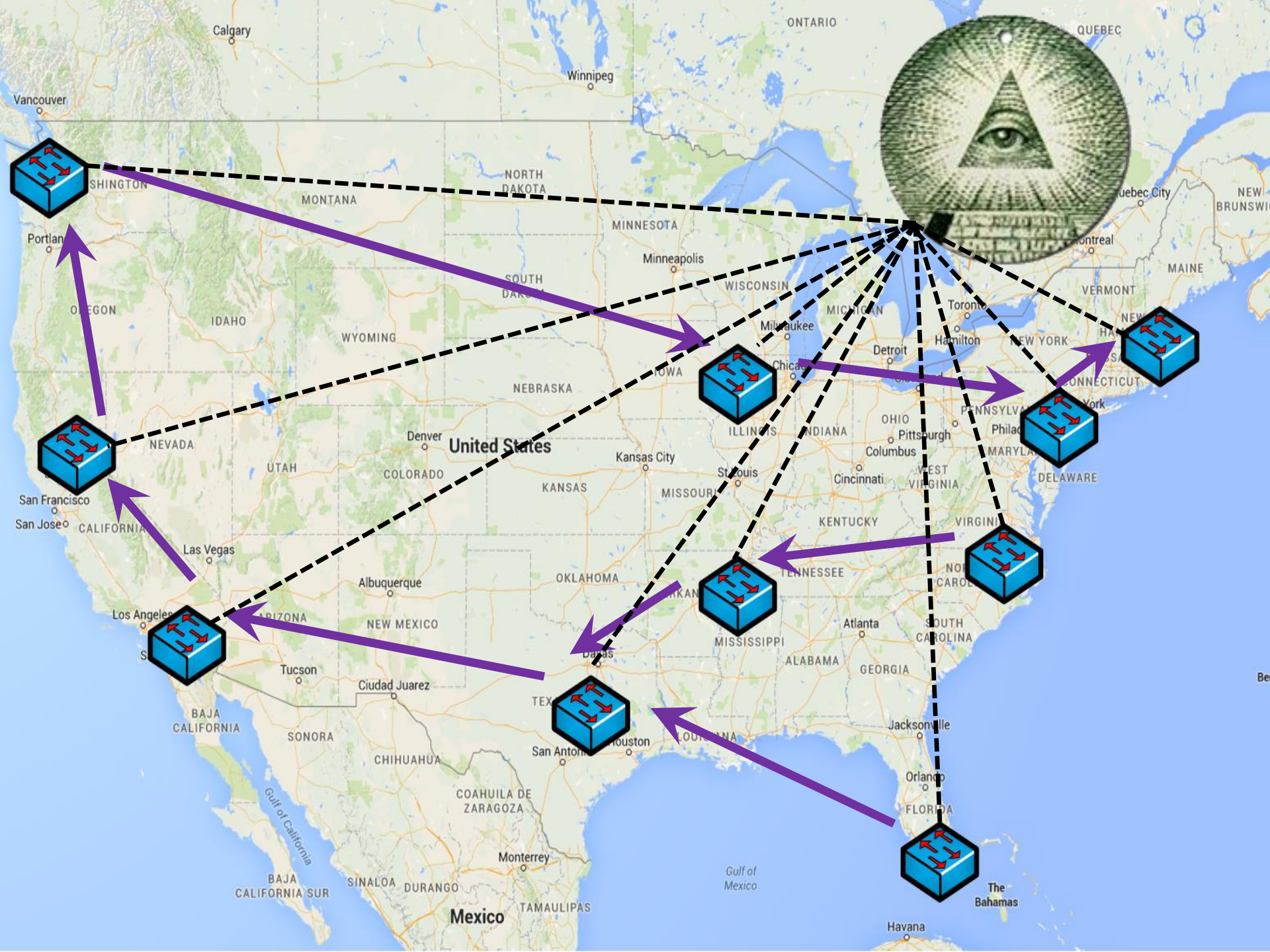






United States

Mexico





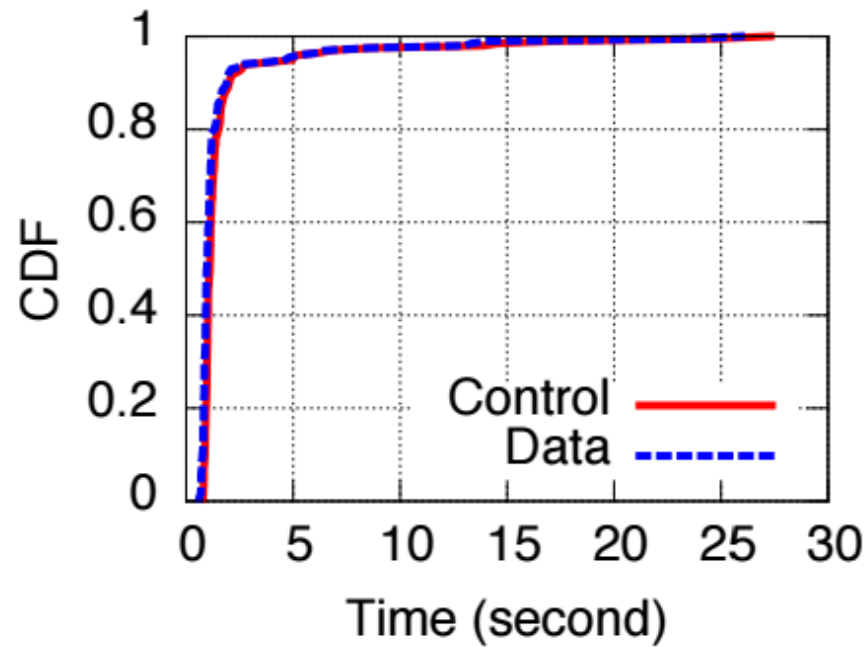








Appears in Practice



*“some switches can ‘straggle,’ taking substantially **more time** than average (e.g., 10-100x) to apply an update”*

Jin et al., SIGCOMM 2014





"Tree Ordering"



"Tree Ordering"



"Tree Ordering"



“Tree Ordering”

Software-Defined Networking



Centralized controller updates networks rules for optimization
Controller (*control plane*) updates the switches/routers (*data plane*)





old network
rules



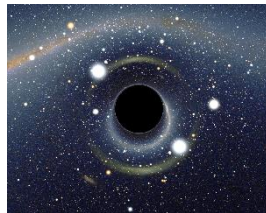
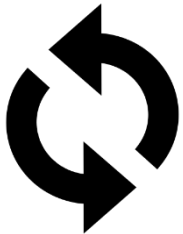
new network
rules



old network rules



new network rules





old network rules



new network rules

possible solution: be fast!



e.g., B4 [Jain et al., 2013]



old network
rules



new network
rules

possible solution: be consistent!

e.g.,

- per-router ordering [Vanbever et al., 2012]
- two phase commit [Reitblatt et al., 2012]
- SWAN [Hong et al., 2013]
- Dionysus [Jin et al., 2014]
-



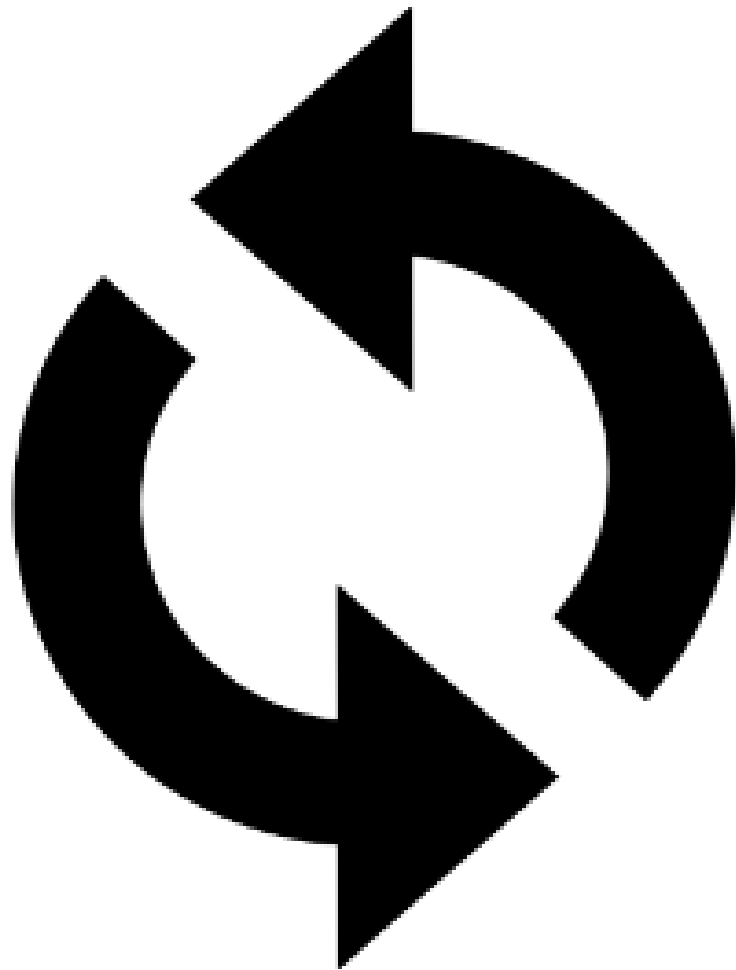
old network rules



new network rules

possible solution: be consistent!

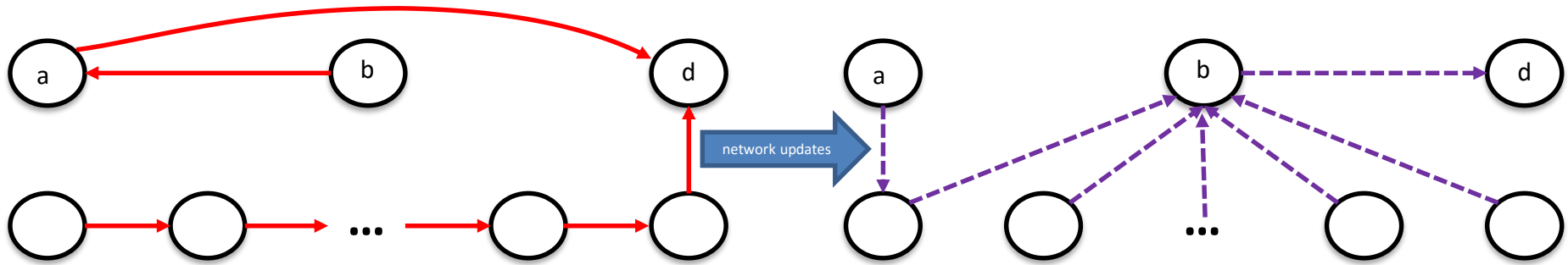




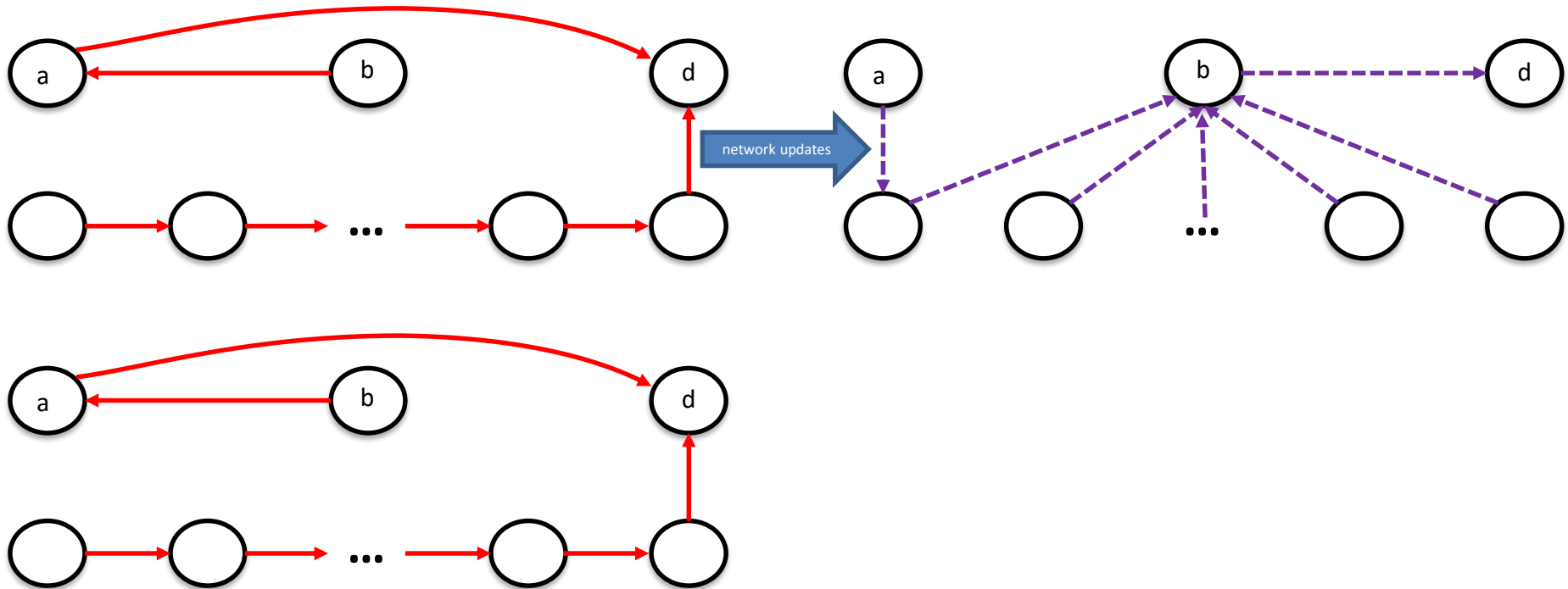
Dynamic Updates

Idea: Update as many routers as you can

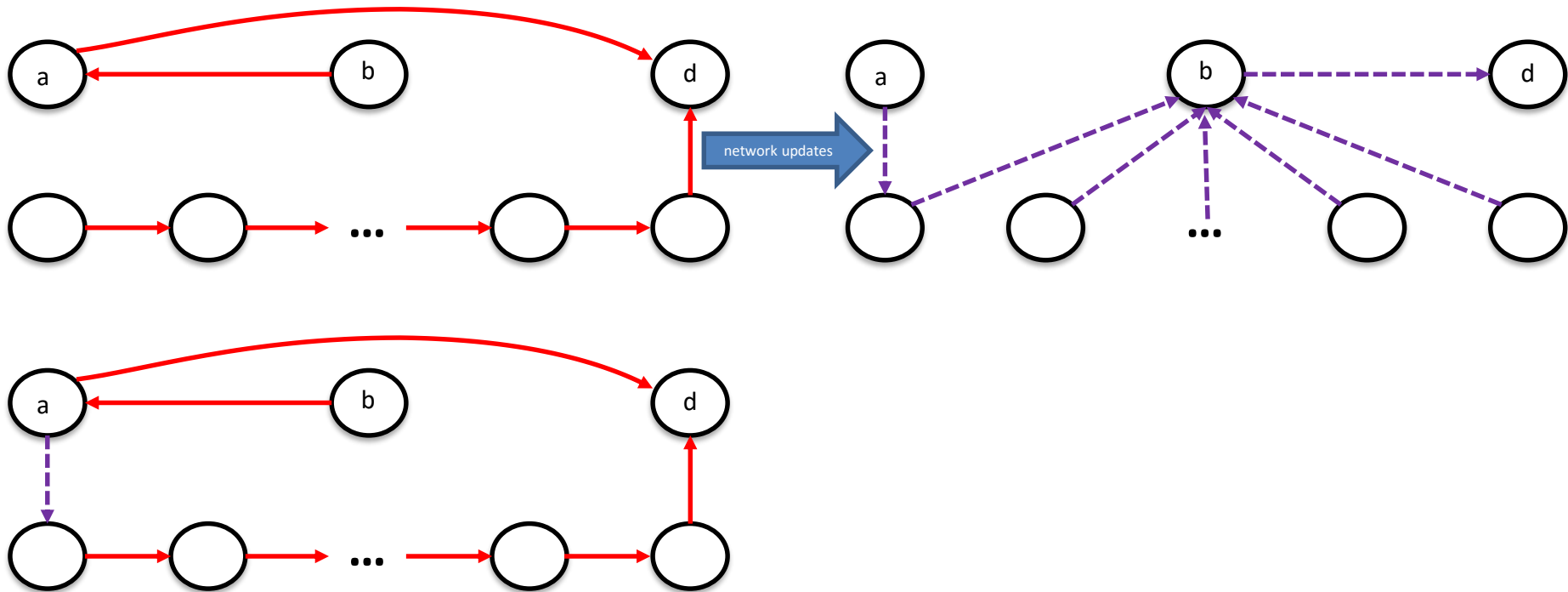
Dynamic Updates



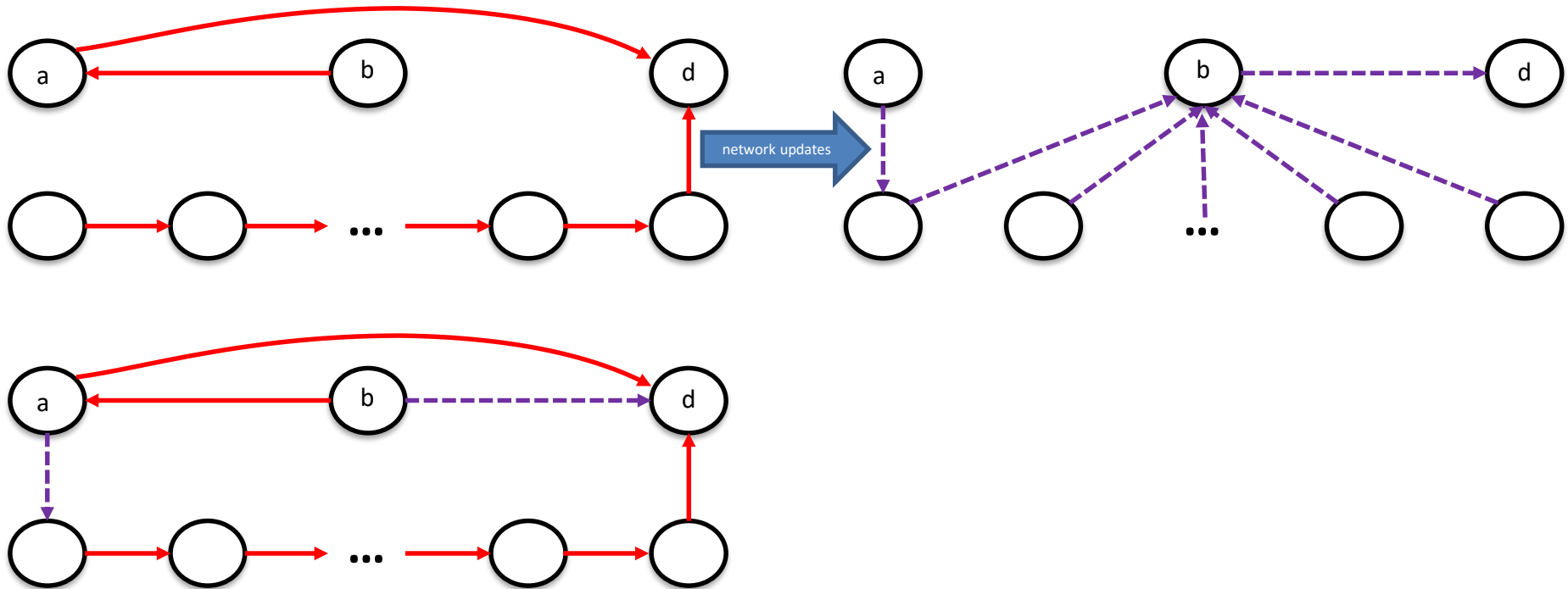
Dynamic Updates



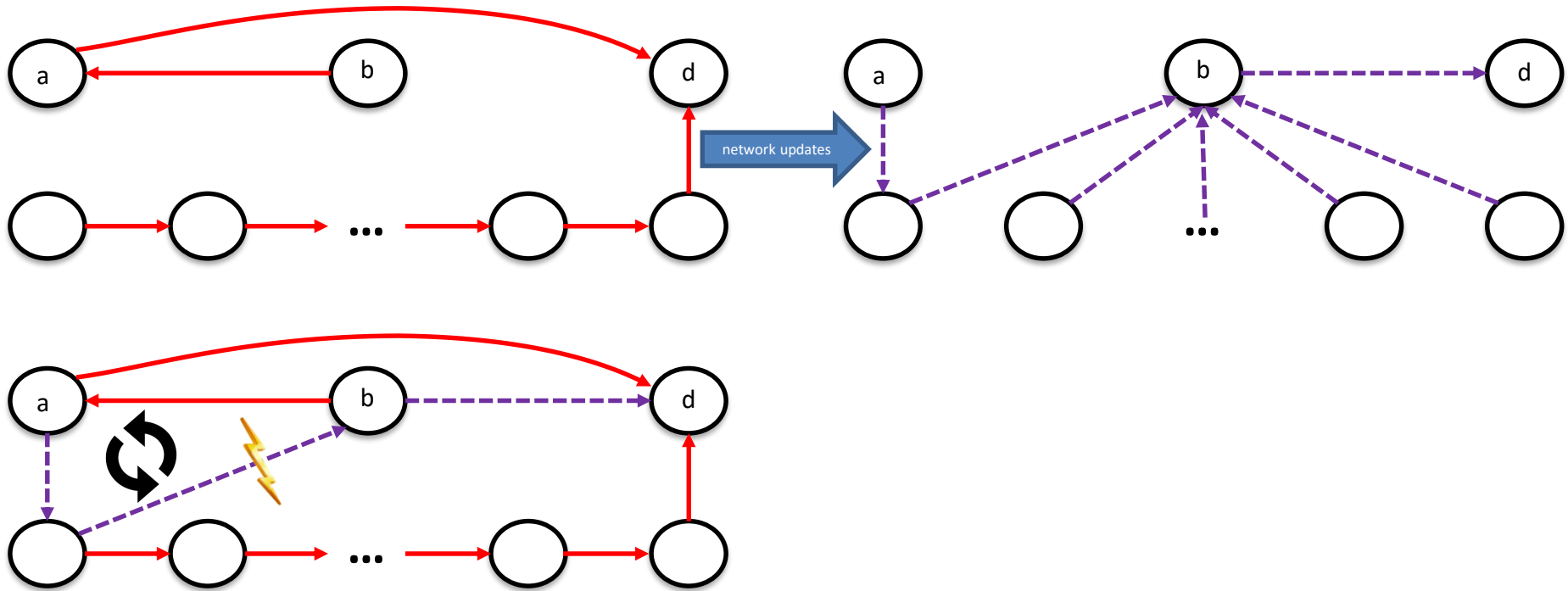
Dynamic Updates



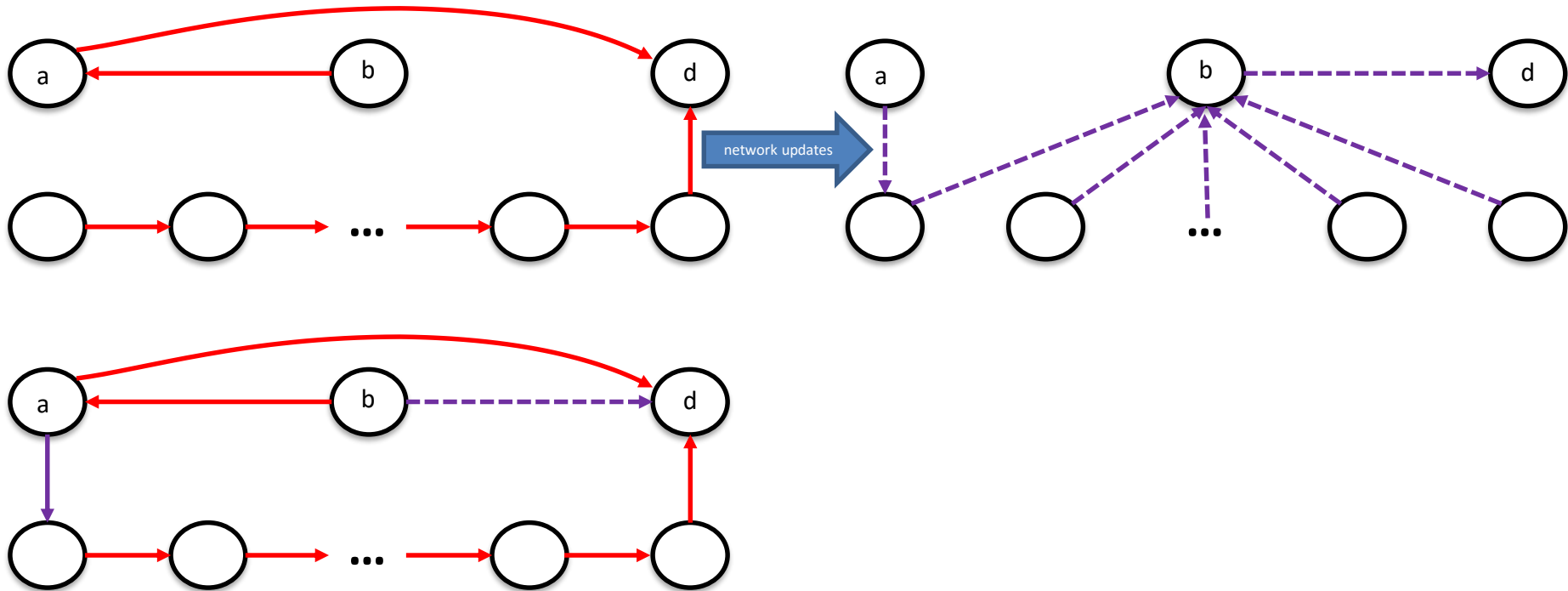
Dynamic Updates



Dynamic Updates

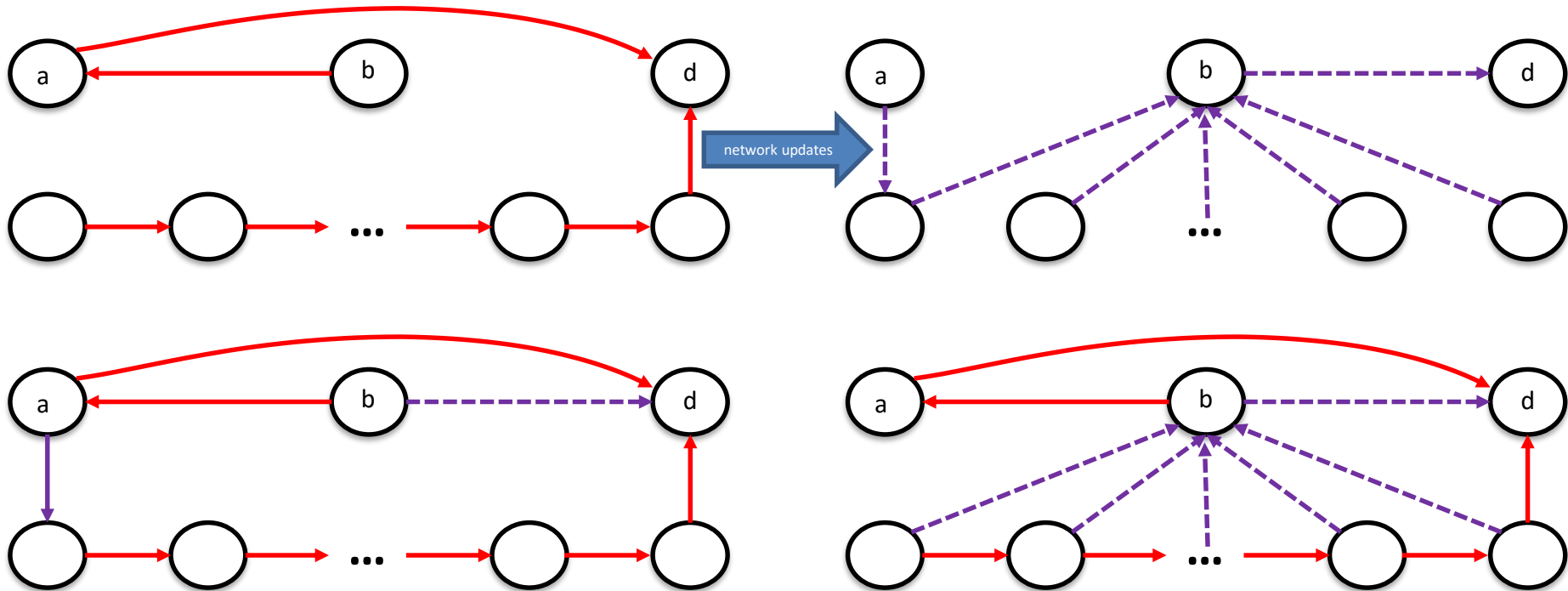


Dynamic Updates



greedy **maximum** update
a & b update → all others wait
2 nodes update

Dynamic Updates



greedy **maximum** update
a & b update → all others wait
2 nodes update

maximal update
a waits → all others update
all but 1 update

Dynamic Updates

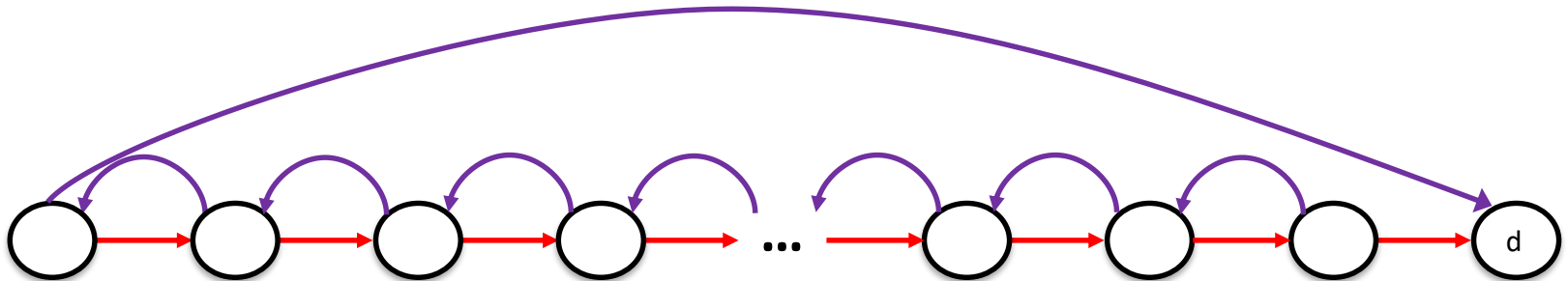
Maximize #routers updated \approx Feedback Arc Set
 \Rightarrow Approximate within $O(\log n \log \log n)$

Dynamic Updates

But how long until all routers updated?

Dynamic Updates vs Tree Ordering

Worst case? Identical to Tree Ordering



Dynamic Updates vs Tree Ordering

But in practice?

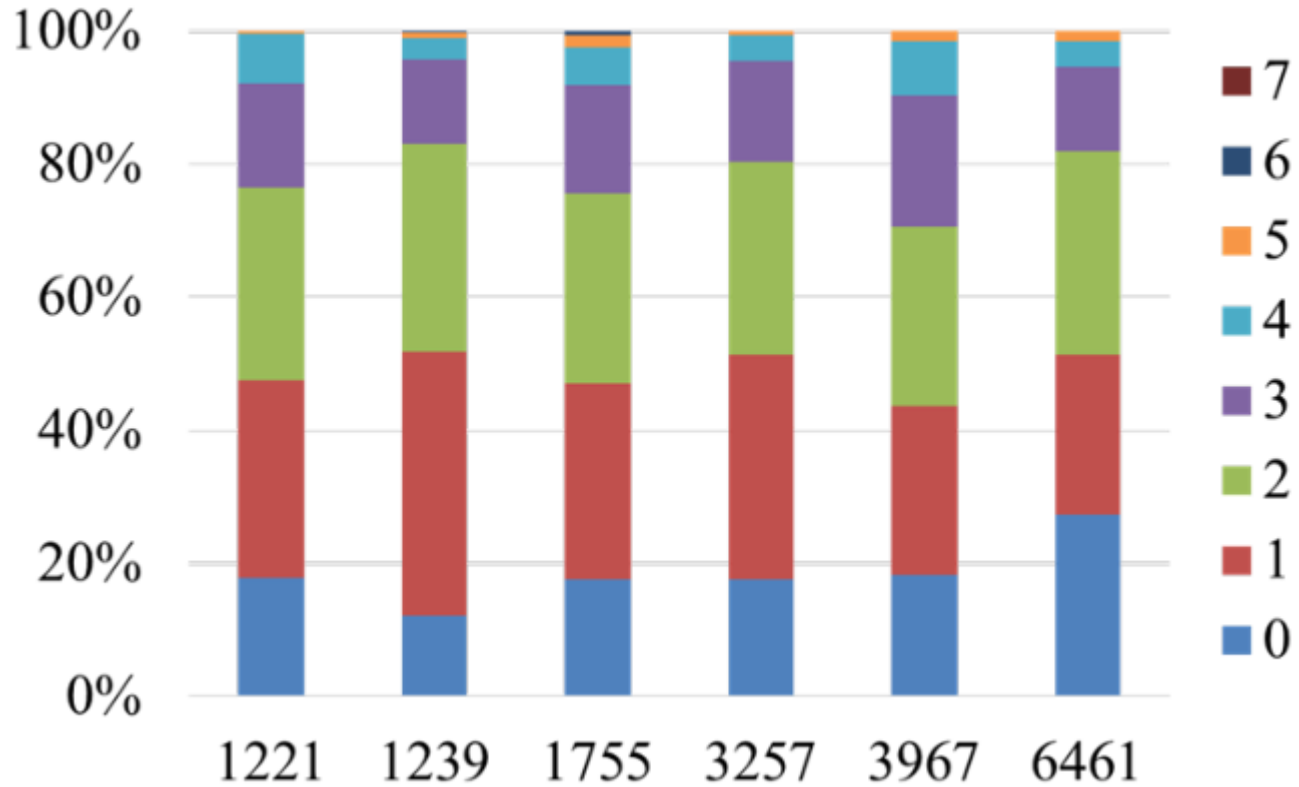
Dynamic updates vs. Tree Ordering ?

#updates on Tier 1 ISP [single-link failure]

Tree Ordering: ≤ 14 updates [FB, 2007]

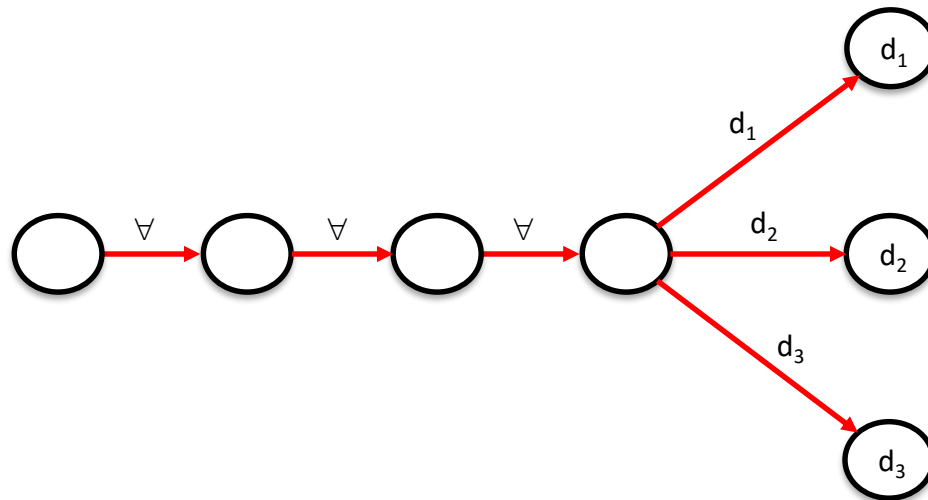
Dynamic Updates: ≤ 7 updates

Dynamic Updates



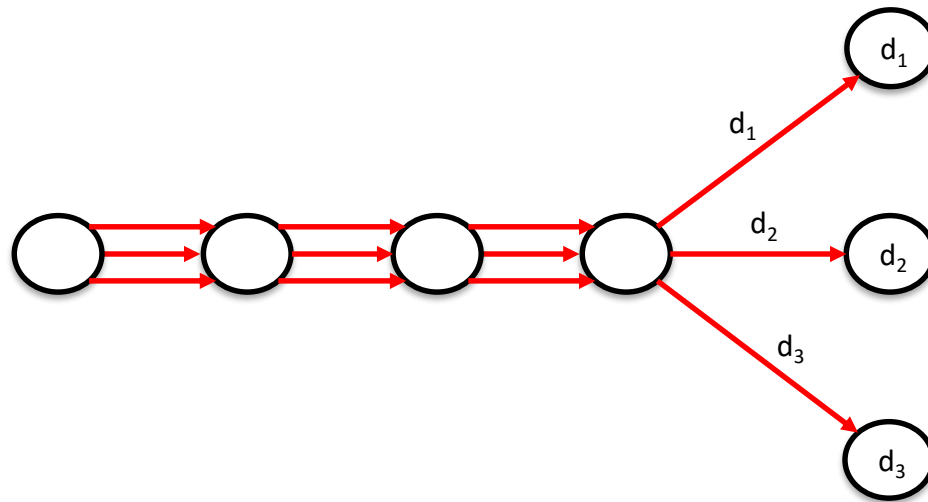
Beyond a Single Destination

But what about prefix routing rules?



Beyond a Single Destination

Split into single destination rules? (Memory...)



Beyond a Single Destination

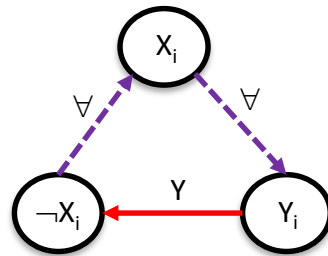
Fewest #updates: NP-hard to approximate

Beyond a Single Destination

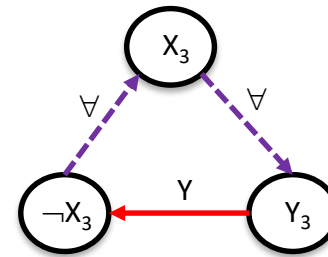
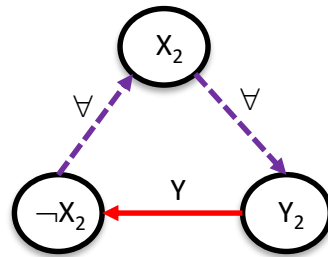
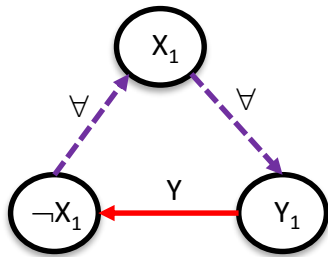
Reduction from 3-Satisfiability

1. Maximizing #rules per update is NP-hard
2. Fewest #updates NP-hard to approximate

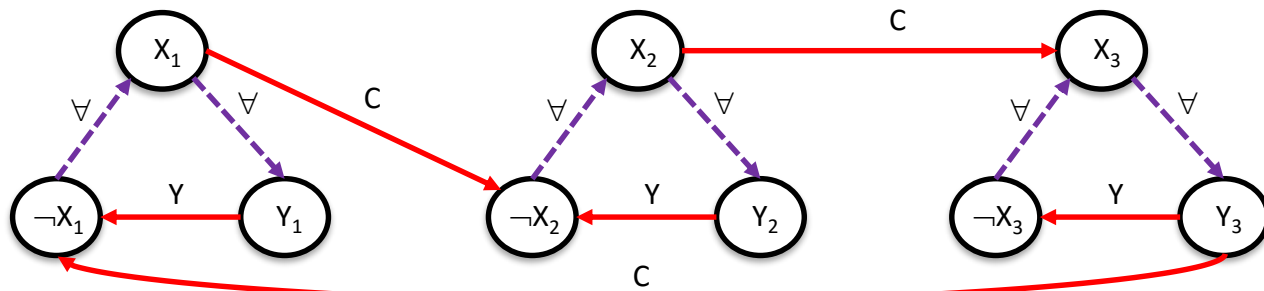
3-Satisfiability



3-Satisfiability

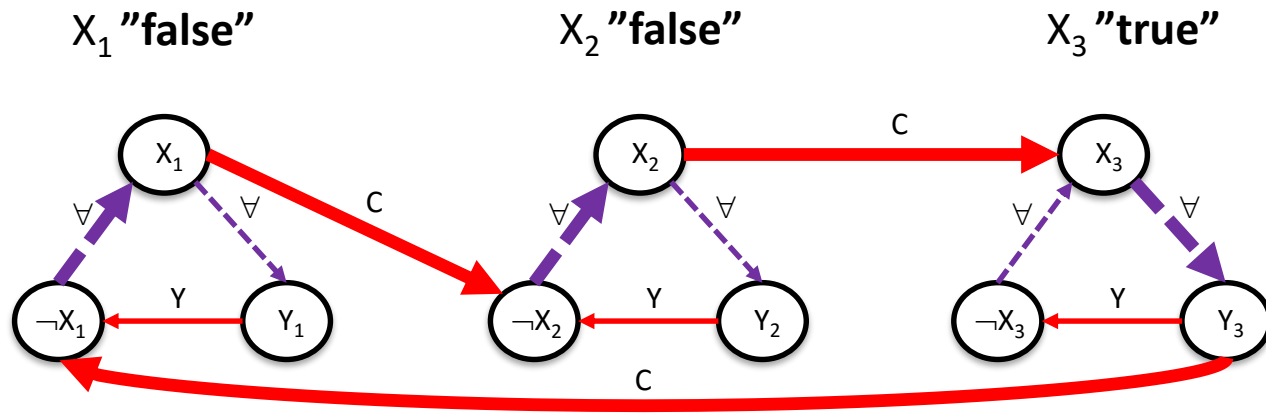


3-Satisfiability



$$C = X_1 \vee X_2 \vee \neg X_3$$

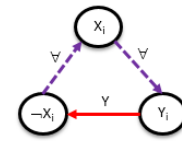
3-Satisfiability



$$C = X_1 \vee X_2 \vee \neg X_3$$

3-Satisfiability

3-Satisfiable \Leftrightarrow Update each variable*



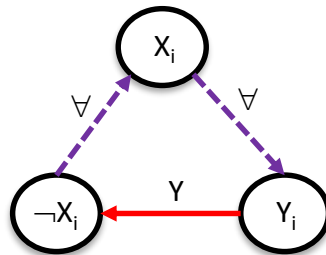
**(and all rules outside the variables)*

Shortest Sequence

Also: Shortest sequence NP-hard to approximate

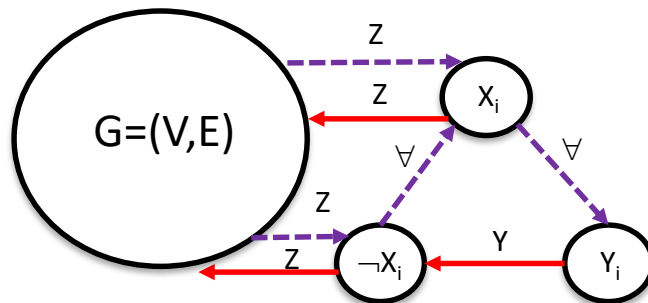
Shortest Sequence

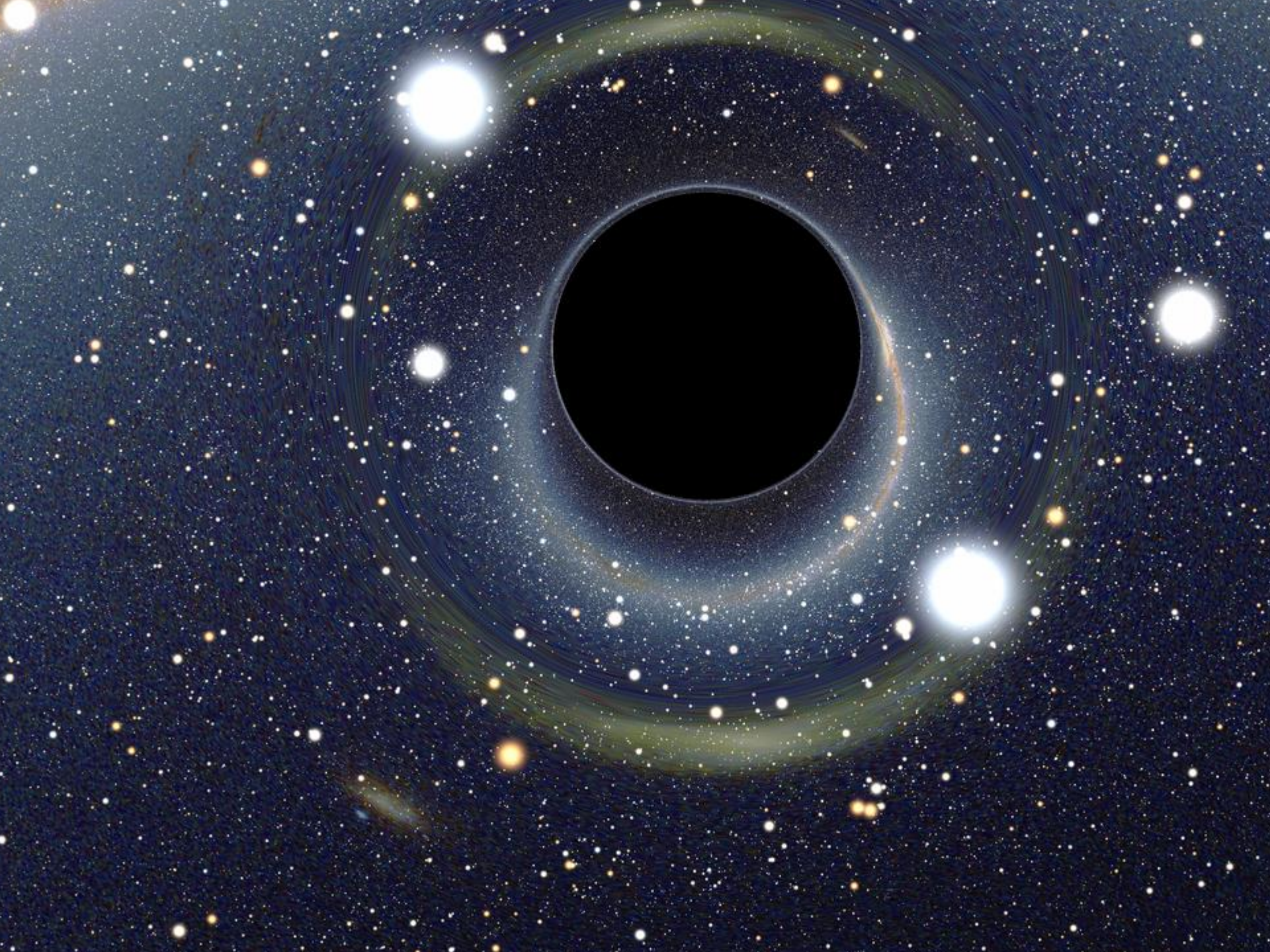
Also: Shortest sequence NP-hard to approximate



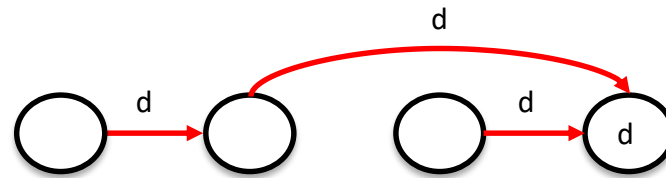
Shortest Sequence

Also: Shortest sequence NP-hard to approximate

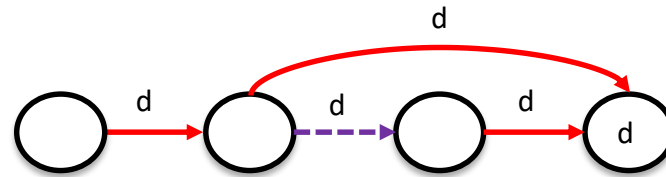




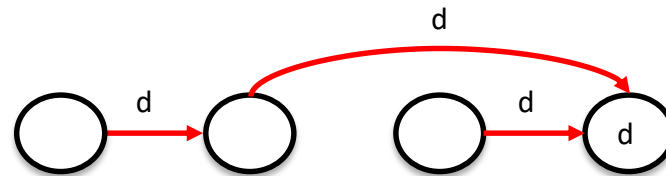
Blackholes



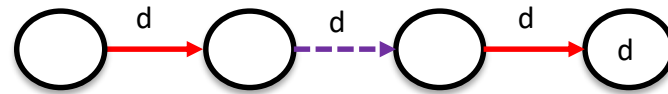
Blackholes



Blackholes

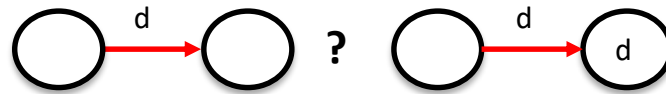


Blackholes



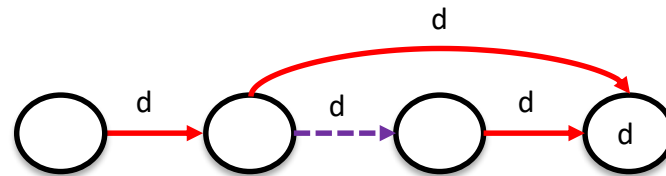
Blackholes

No rule? Drop packet



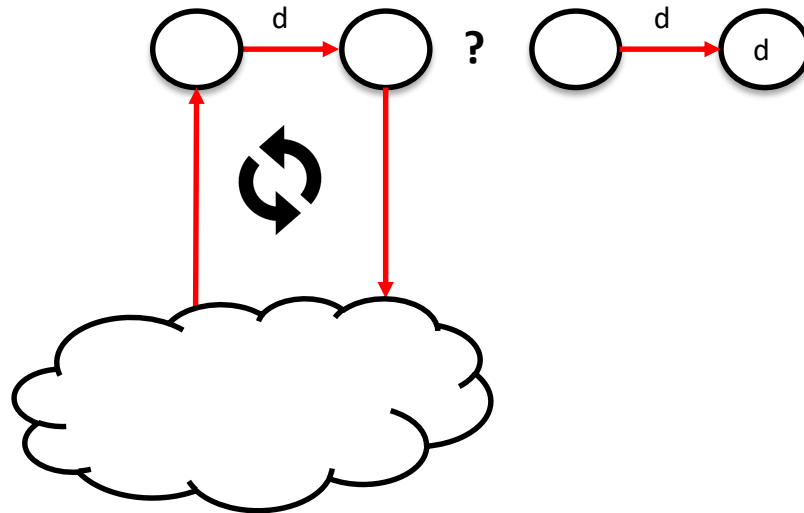
Blackholes

Idea 1: Keep old rule in memory



Blackholes

Idea 2: Send somewhere else?



Blackholes

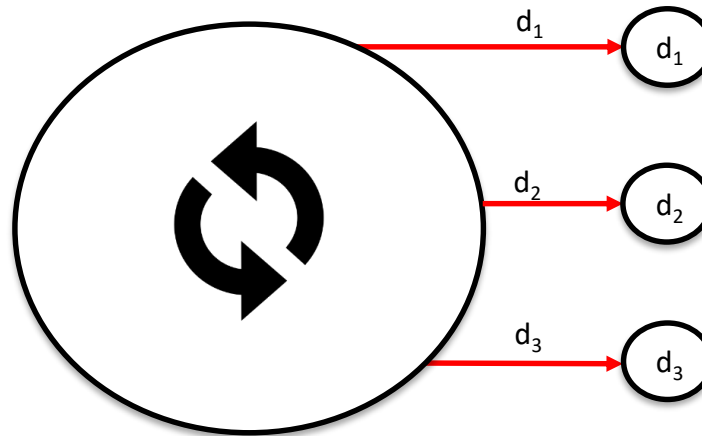
Respect memory limits and no loops?

Blackholes

Fastest method: NP-hard to approximate!

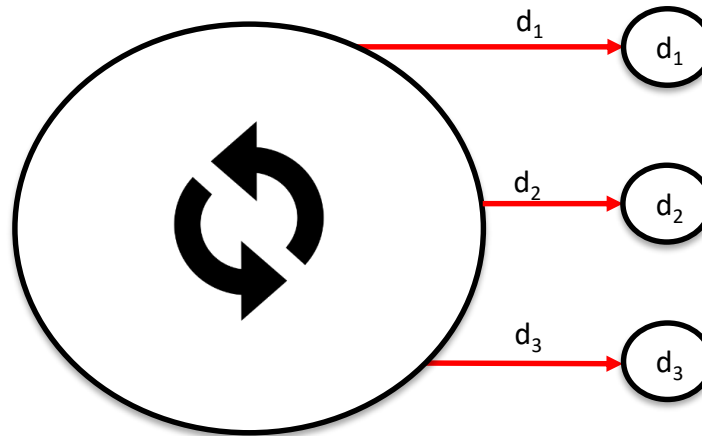
Proof Idea

A cycle can be used for loop-free routing



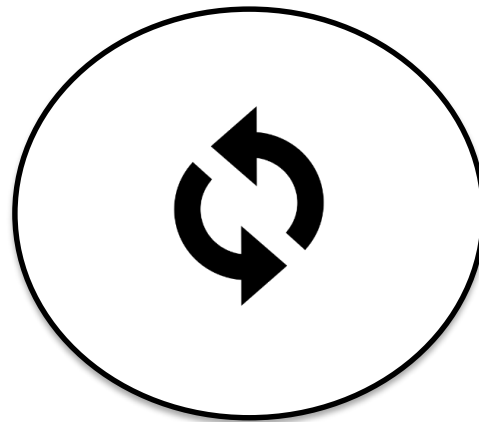
Proof Idea

With cycle: fast change; else slow...

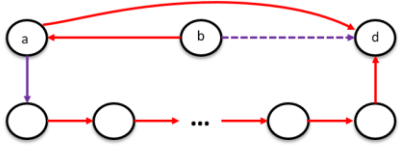


Proof Idea

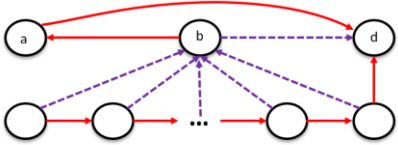
But Hamiltonian Cycle problem is NP-complete!



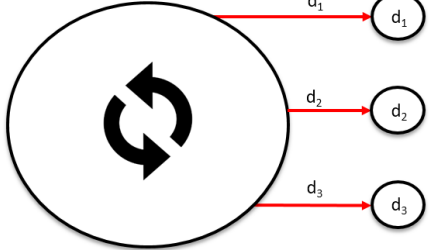
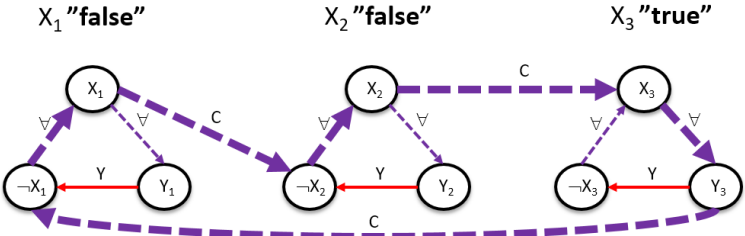
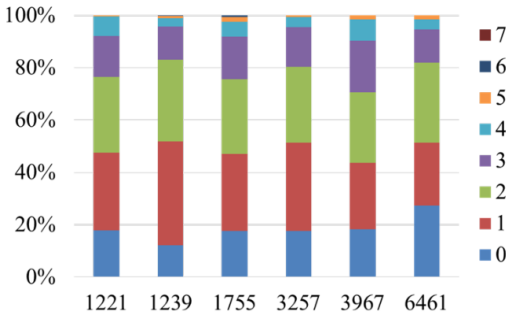
Conclusion



greedy **maximum** update
 a & b update → all others wait
2 nodes update



maximal update
 a waits → all others update
all but 1 update



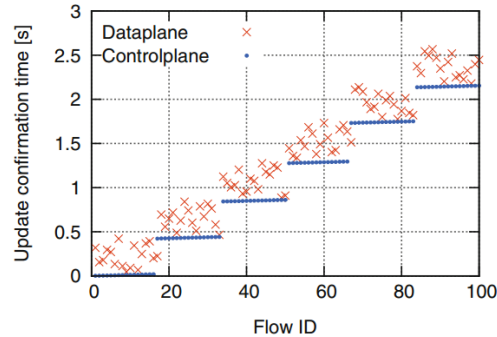
Consistent Updates in Software Defined Networks: On Dependencies, Loop Freedom, and Blackholes



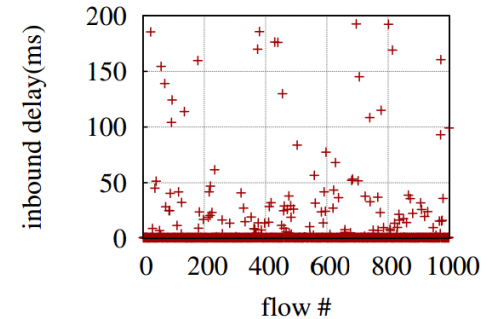
Klaus-Tycho Förster, Ratul Mahajan, and Roger Wattenhofer



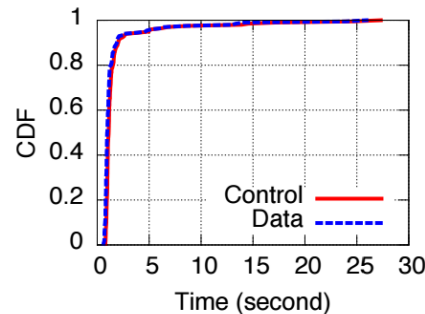
Appears in Practice



“Data plane **updates may fall behind** the control plane acknowledgments and may be even **reordered.**”
Kuzniar et al., PAM 2015

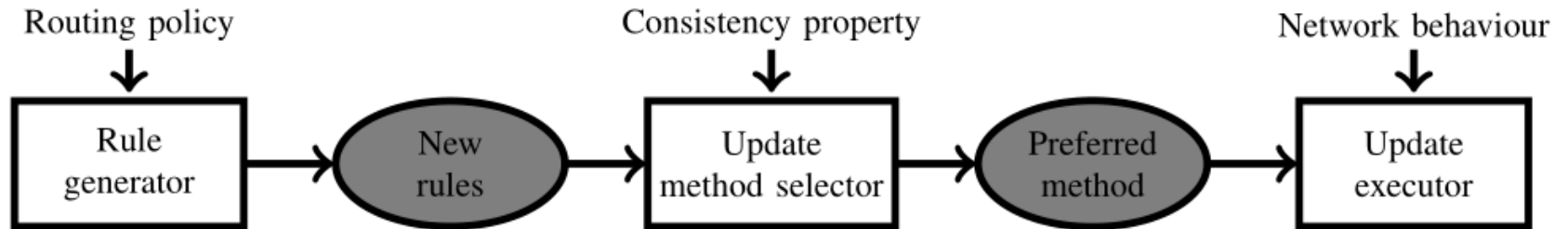


“...the inbound latency is **quite variable** with a [...] standard deviation of 31.34ms...”
He et al., SOSR 2015



“some switches can ‘**straggle,**’ taking substantially **more time** than average (e.g., 10-100x) to apply an update”
Jin et al., SIGCOMM 2014

How to proceed in practice?



Consistent Updates in Software Defined Networks: On Dependencies, Loop Freedom, and Blackholes



Klaus-Tycho Förster, Ratul Mahajan, and Roger Wattenhofer