ETH zürich

Networked Systems
ETH Zürich — seit 2015

Snowcap: Synthesizing Network-Wide Configuration Updates

Tibor Schneider Rüdiger Birkner Laurent Vanbever

SIGCOMM'21, August 24, 2021

Network reconfigurations happen often

Daily Routing policy adaptions[1]

Monthly Traffic engineering adjustments[1]

Yearly Major network redesign[2]

[1] Stefano Vissicchio et al. "Improving Network Agility with Seamless BGP Reconfigurations". In: *IEEE/ACM Transactions on Networking*. 2012

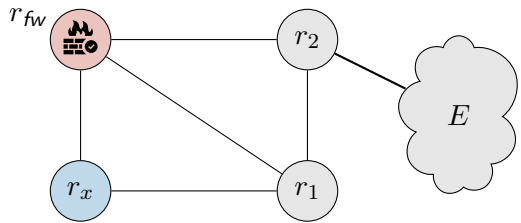
[2] Arjun Singh et al. "Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network". In: *ACM SIGCOMM*. 2015.

Network reconfigurations happen often and cause incidents

Alibaba revealed that **56%** of the incidents
are caused by configuration updates[3]

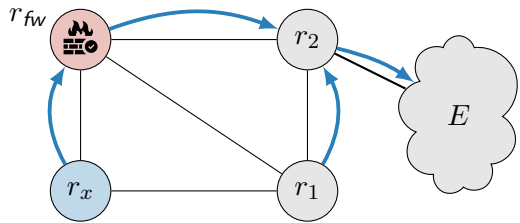
[3] Hongqiang Harry Liu et al. "Automatic Life Cycle Management of Network Configurations". In: *ACM SelfDN*. 2018.

Careless migration can cause unallowed and unnecessary traffic shifts



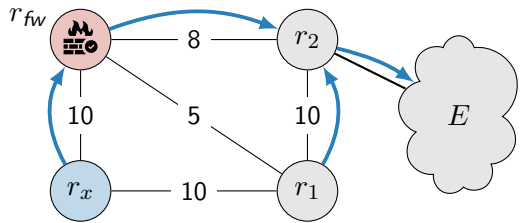
Objective ϕ:
Traffic from r_x to E
must traverse r_{fw}.

Careless migration can cause unallowed and unnecessary traffic shifts



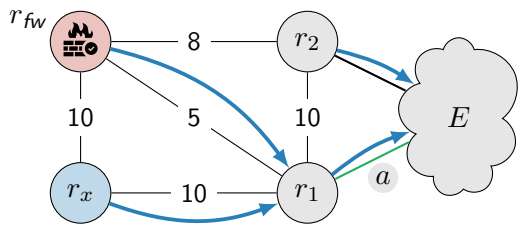
Objective ϕ:
Traffic from r_x to E
must traverse r_{fw}.

Careless migration can cause unallowed and unnecessary traffic shifts



Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

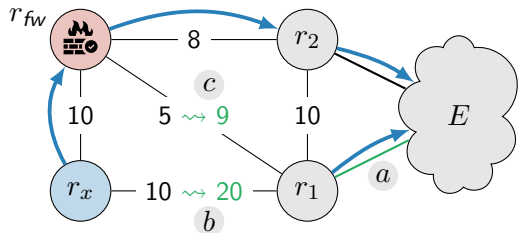
Careless migration can cause unallowed and unnecessary traffic shifts



Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

a eBGP session: $E \longrightarrow r_1$

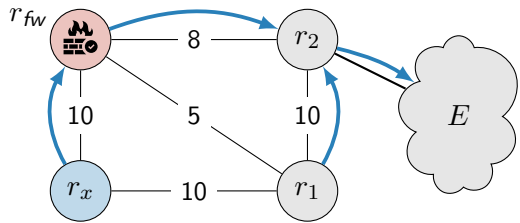
Careless migration can cause unallowed and unnecessary traffic shifts



Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

- a eBGP session: $E \longrightarrow r_1$
- b Link weight r_x — r_1: 10 ⇝ 20
- c Link weight r_{fw} — r_1: 5 ⇝ 9

Careless migration can cause unallowed and unnecessary traffic shifts



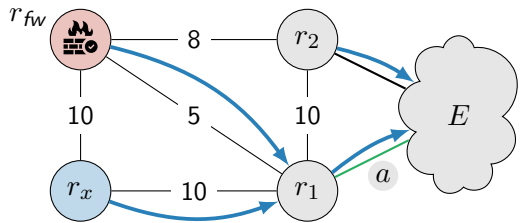
Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

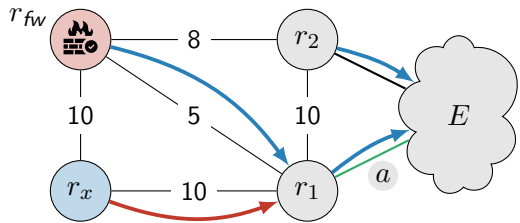
Careless migration can cause unallowed and unnecessary traffic shifts



Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

- a eBGP session: $E \longrightarrow r_1$
- b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20
- c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Careless migration can cause unallowed and unnecessary traffic shifts



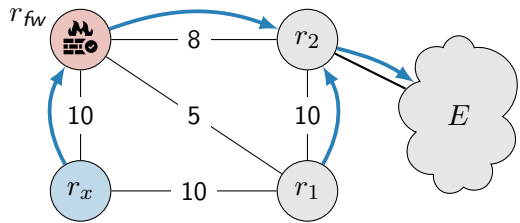
Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

(a) eBGP session: $E \longrightarrow r_1$

(b) Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

(c) Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Careless migration can cause unallowed and unnecessary traffic shifts



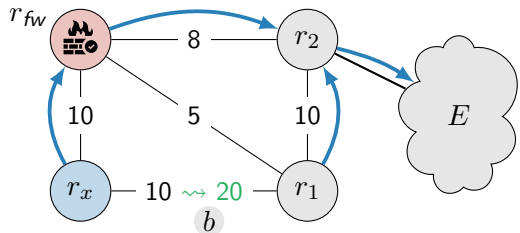
Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Careless migration can cause unallowed and unnecessary traffic shifts



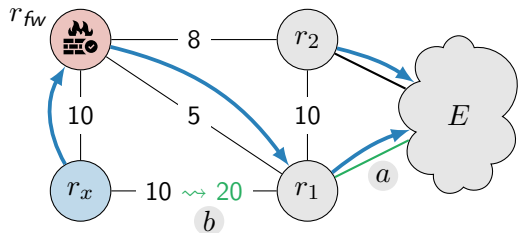
Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 ⇝ 20

c Link weight r_{fw} — r_1: 5 ⇝ 9

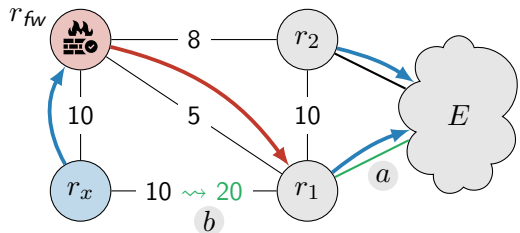
Careless migration can cause unallowed and unnecessary traffic shifts



Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

- a eBGP session: $E \longrightarrow r_1$
- b Link weight r_x — r_1: 10 ⇝ 20
- c Link weight r_{fw} — r_1: 5 ⇝ 9

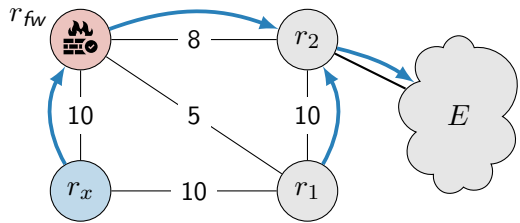
Careless migration can cause unallowed and unnecessary traffic shifts



Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

- a eBGP session: $E \longrightarrow r_1$
- b Link weight r_x — r_1: 10 ⇝ 20
- c Link weight r_{fw} — r_1: 5 ⇝ 9

Careless migration can cause unallowed and unnecessary traffic shifts



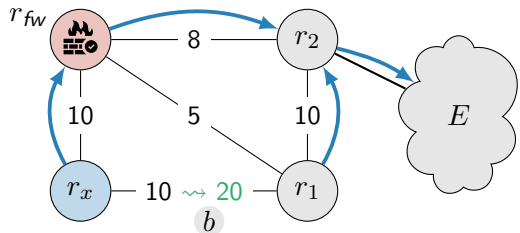
Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Careless migration can cause unallowed and unnecessary traffic shifts



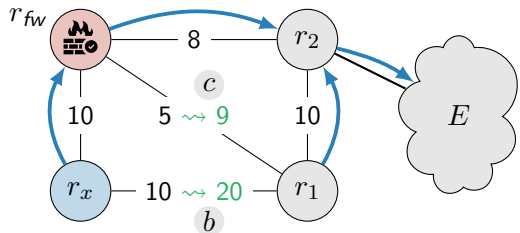
Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 ⇝ 20

c Link weight r_{fw} — r_1: 5 ⇝ 9

Careless migration can cause unallowed and unnecessary traffic shifts



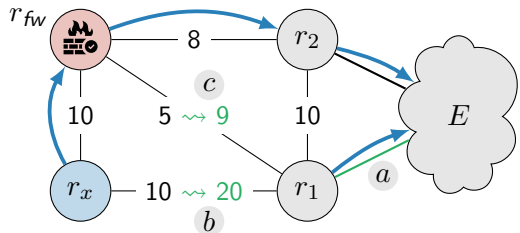
Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 ⇝ 20

c Link weight r_{fw} — r_1: 5 ⇝ 9

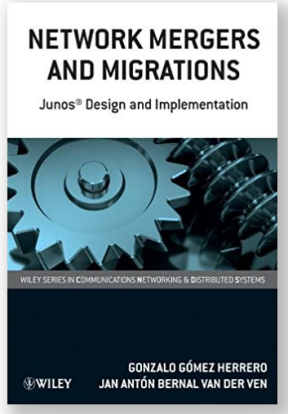
Careless migration can cause unallowed and unnecessary traffic shifts



Objective ϕ:
Traffic from r_x to E *must* traverse r_{fw}.

- a eBGP session: $E \longrightarrow r_1$
- b Link weight r_x — r_1: 10 ⇝ 20
- c Link weight r_{fw} — r_1: 5 ⇝ 9

A few best practices exist



Best practices are not enough

Best practices are not enough

Migration from iBGP full-mesh to route-reflection:

$\geq 50\%$ chance to violate reachability

Best practices are not enough

Migration from iBGP full-mesh to route-reflection:

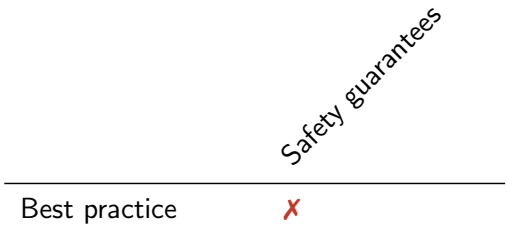
$\geq 50\%$ chance to violate reachability	
Random order	$\mathbf{70}\%$

Best practices are not enough

Migration from iBGP full-mesh to route-reflection:

$\geq 50\%$ chance to violate reachability	
Random order	$\mathbf{70}\%$
Best practice order	$\mathbf{25}\%$

Existing systems are limited



Existing systems are limited

	Safety guarantees	Any protocols
Best practice	✗	✗

Existing systems are limited

	Safety guarantees	Any protocols	Any scenarios
Best practice	✗	✗	✗

Existing systems are limited

	Safety guarantees	Any protocols	Any scenarios
Best practice	✗	✗	✗
"Offline" systems [4]	✓	✓	✓

[4] Laurent Vanbever et al. "Lossless Migrations of Link-State IGPs". In: *IEEE/ACM Transactions on Networking*. 2012
Stefano Vissicchio et al. "Improving Network Agility with Seamless BGP Reconfigurations". In: *IEEE/ACM Transactions on Networking*. 2012

Existing systems are limited

	Safety guarantees	Any protocols	Any scenarios	Low overhead
Best practice	✗	✗	✗	✓
"Offline" systems [4]	✓	✓	✓	✗

[4] Laurent Vanbever et al. "Lossless Migrations of Link-State IGPs". In: *IEEE/ACM Transactions on Networking*. 2012
Stefano Vissicchio et al. "Improving Network Agility with Seamless BGP Reconfigurations". In: *IEEE/ACM Transactions on Networking*. 2012

Existing systems are limited

	Safety guarantees	Any protocols	Any scenarios	Low overhead
Best practice	✗	✗	✗	✓
"Offline" systems [4]	✓	✓	✓	✗
"Online" systems [5]	✓	✗	✗	✓

[4] Laurent Vanbever et al. "Lossless Migrations of Link-State IGPs". In: *IEEE/ACM Transactions on Networking*. 2012
Stefano Vissicchio et al. "Improving Network Agility with Seamless BGP Reconfigurations". In: *IEEE/ACM Transactions on Networking*. 2012

[5] Pierre Francois et al. "Avoiding Transient Loops During the Convergence of Link-State Routing Protocols". In: *IEEE/ACM Transactions on Networking*. 2007
Pierre Francois et al. "Avoiding Disruptions during Maintenance Operations on BGP Sessions". In: *IEEE Transactions on Network and Service Management*. 2007

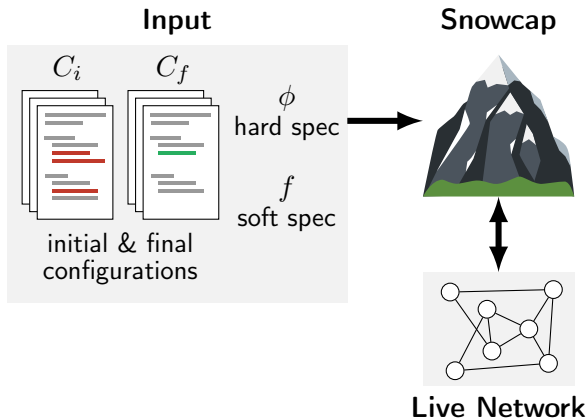
Existing systems are limited

	Safety guarantees	Any protocols	Any scenarios	Low overhead
Best practice	✗	✗	✗	✓
"Offline" systems [4]	✓	✓	✓	✗
"Online" systems [5]	✓	✗	✗	✓
Snowcap	✓	✓	✓	✓

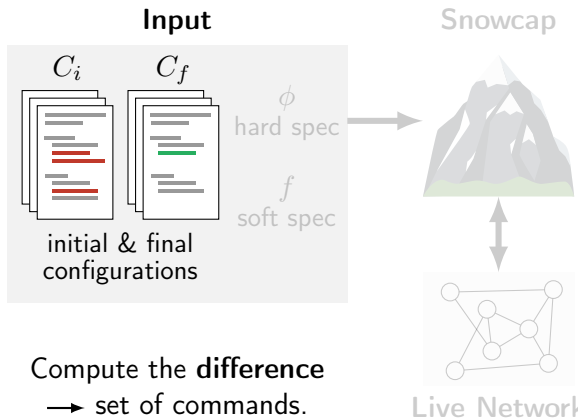
[4] Laurent Vanbever et al. "Lossless Migrations of Link-State IGPs". In: *IEEE/ACM Transactions on Networking*. 2012
Stefano Vissicchio et al. "Improving Network Agility with Seamless BGP Reconfigurations". In: *IEEE/ACM Transactions on Networking*. 2012

[5] Pierre Francois et al. "Avoiding Transient Loops During the Convergence of Link-State Routing Protocols". In: *IEEE/ACM Transactions on Networking*. 2007
Pierre Francois et al. "Avoiding Disruptions during Maintenance Operations on BGP Sessions". In: *IEEE Transactions on Network and Service Management*. 2007

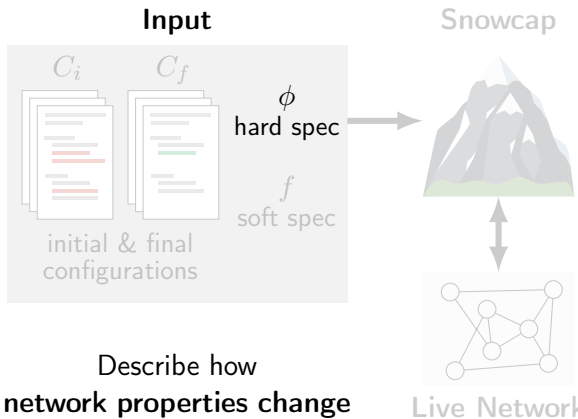
Snowcap performs network reconfigurations automatically and safely



Snowcap performs network reconfigurations automatically and safely



Snowcap performs network reconfigurations automatically and safely



Snowcap's specification language is extremely flexible

- **Basic Policies** for each flow:
  Reachability, isolation, redundancy, waypointing.
- **Linear Temporal Logic (LTL)**:
  Express how the policy changes during migration.

Linear Temporal Logic captures policy changes

Firewall migration from r_1 to r_2

$$\left(\bigwedge_{x \in Flows} r_1 \in path_x \ \mathbf{U}\mathbf{G} \ r_2 \in path_x \right)$$

All flows *can* switch at **different** times.

Linear Temporal Logic captures policy changes

Firewall migration from r_1 to r_2

$$\left(\bigwedge_{x \in Flows} r_1 \in path_x \ \mathbf{UXG} \ r_2 \in path_x \right)$$

All flows *are allowed* to bypass the firewall for a short time.

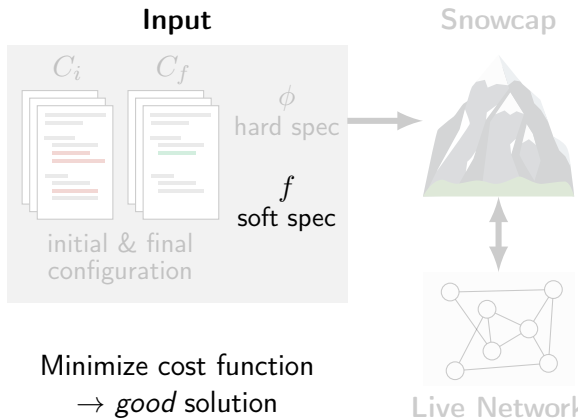
Linear Temporal Logic captures policy changes

Firewall migration from r_1 to r_2

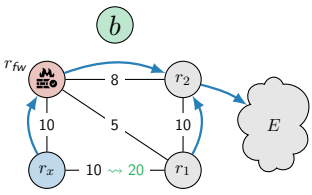
$$\left(\bigwedge_{x \in Flows} r_1 \in path_x \right) \mathbf{UG} \left(\bigwedge_{x \in Flows} r_2 \in path_x \right)$$

All flows *must* change at **the same** time.

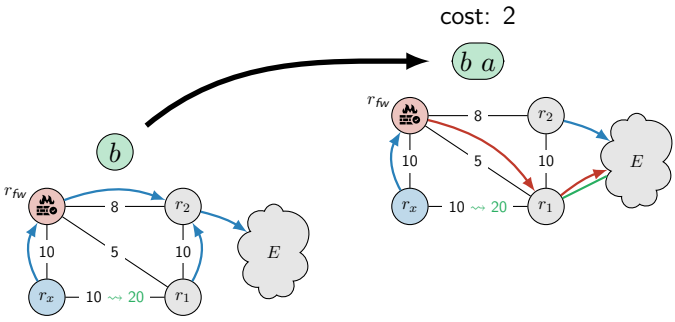
Snowcap performs network reconfigurations automatically and safely



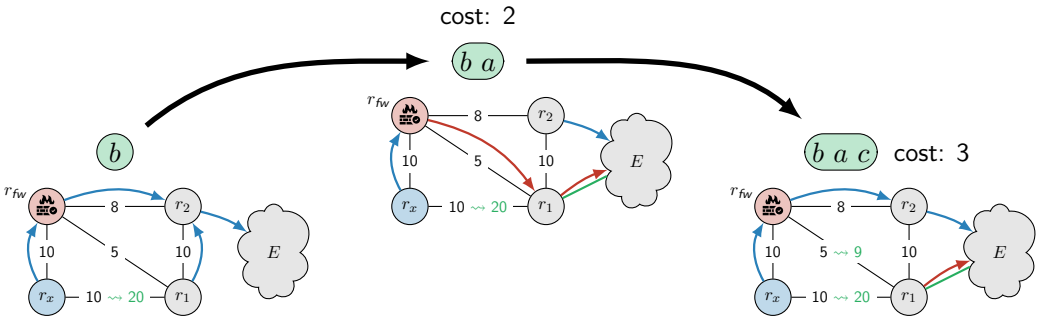
Not all correct orderings are created equally



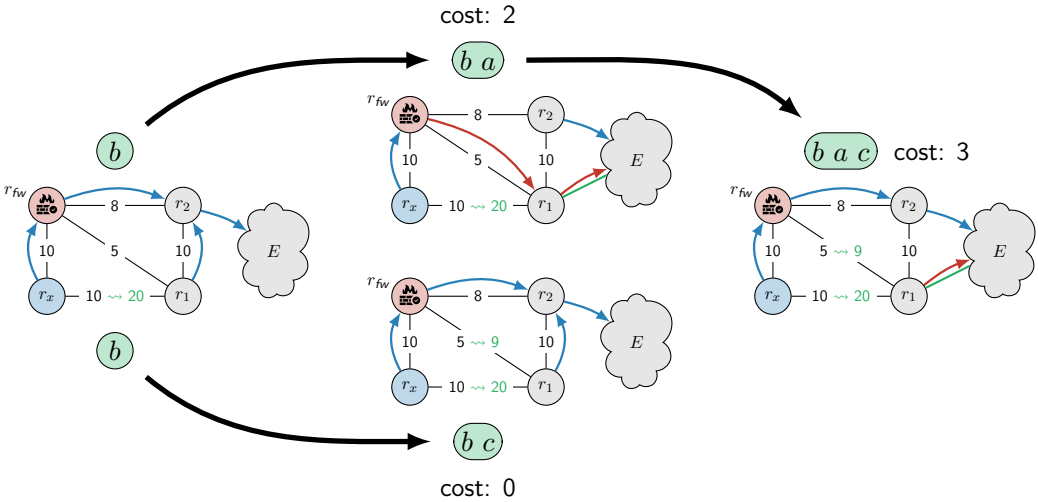
Not all correct orderings are created equally



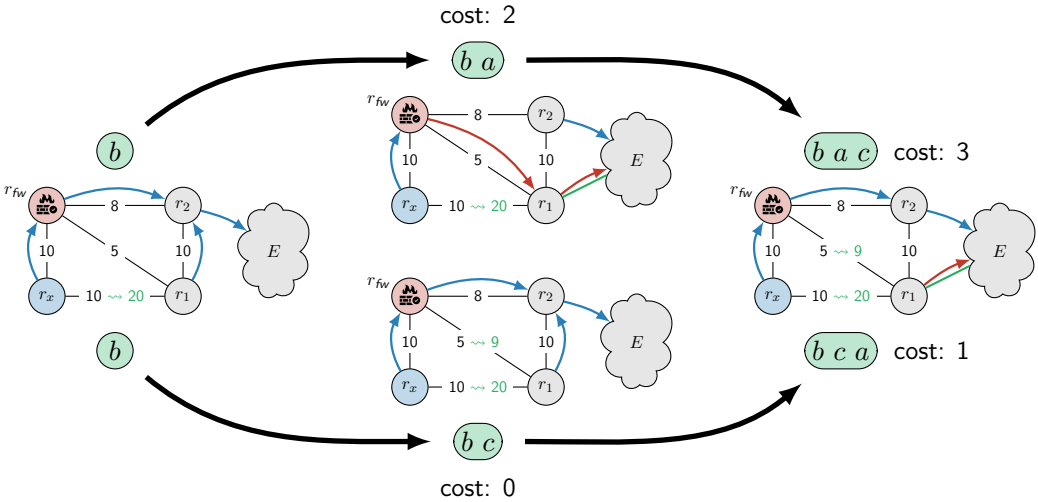
Not all correct orderings are created equally



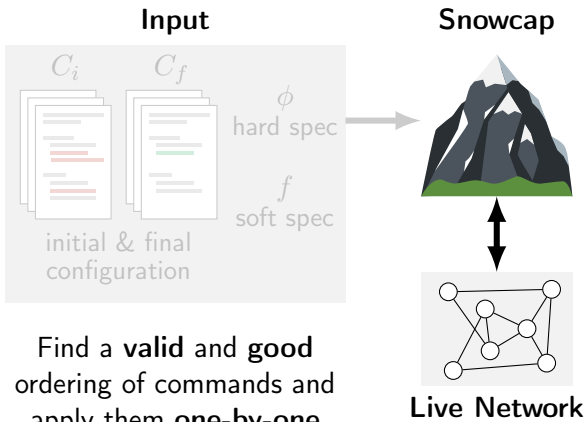
Not all correct orderings are created equally



Not all correct orderings are created equally



Snowcap performs network reconfigurations automatically and safely

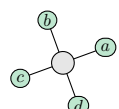


Find a **valid** and **good** ordering of commands and apply them **one-by-one**

It's all about navigating the search space of possible reconfiguration orderings

The search space is both

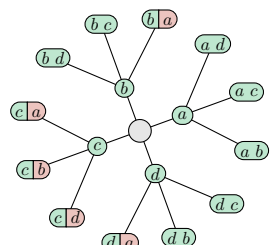
- **sparse**; and
- **huge**.



It's all about navigating the search space of possible reconfiguration orderings

The search space is both

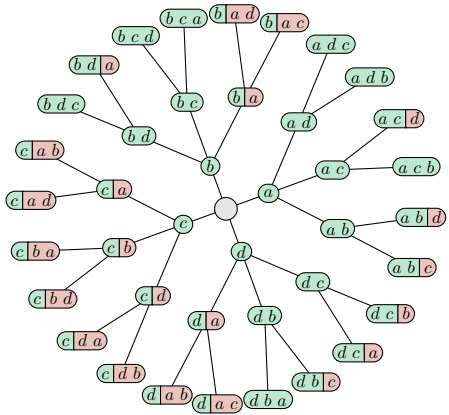
- **sparse**; and
- **huge**.



It's all about navigating the search space of possible reconfiguration orderings

The search space is both

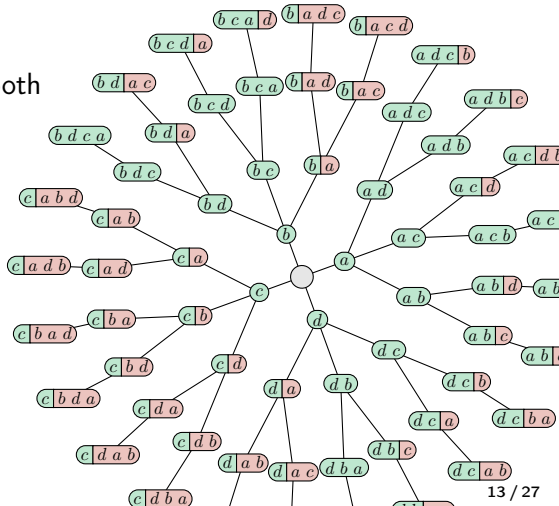
- **sparse**; and
- **huge**.



It's all about navigating the search space of possible reconfiguration orderings

The search space is both

- **sparse**; and
- **huge**.



1. **Exploration**
   How does Snowcap solve simple scenarios?

2. **Counter-example-guided search**
   How does Snowcap solve more difficult scenarios?

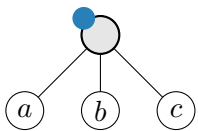
3. **Evaluation**
   How efficient and effective is Snowcap?

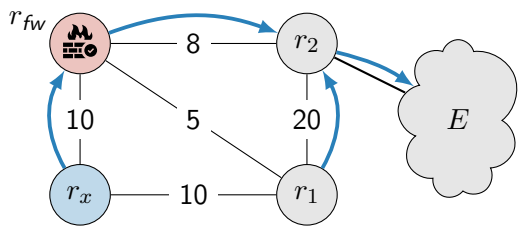
1. **Exploration**
   How does Snowcap solve simple scenarios?

2. **Counter-example-guided search**
   How does Snowcap solve more difficult scenarios?

3. **Evaluation**
   How efficient and effective is Snowcap?

The exploration algorithm is based on DFS traversal



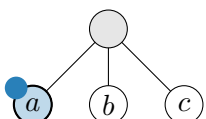


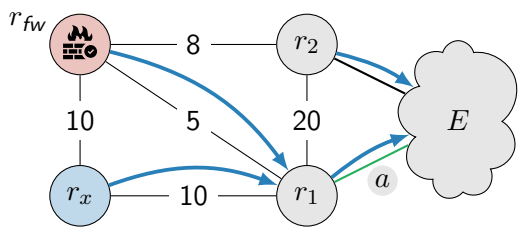
(a) eBGP session: $E \longrightarrow r_1$

(b) Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

(c) Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

The exploration algorithm is based on DFS traversal



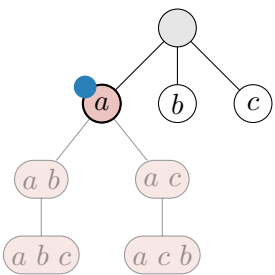


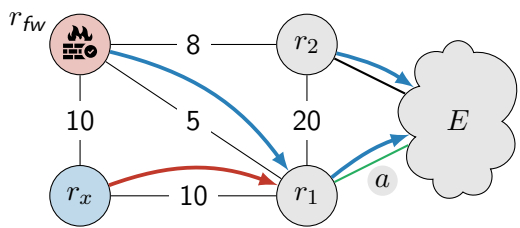
a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Sequences with a known, bad prefix are not explored



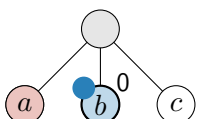


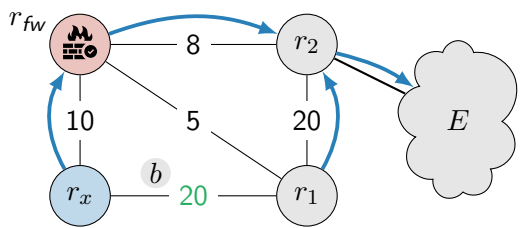
a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Greedy minimization of the cost function



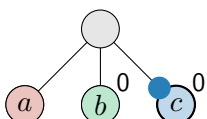


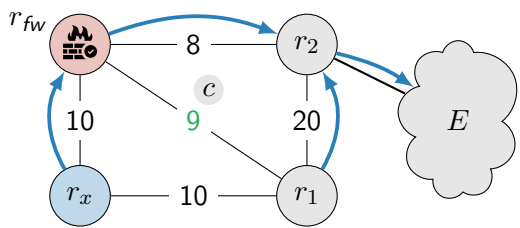
a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Greedy minimization of the cost function



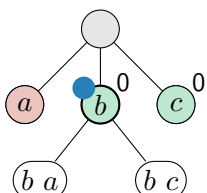


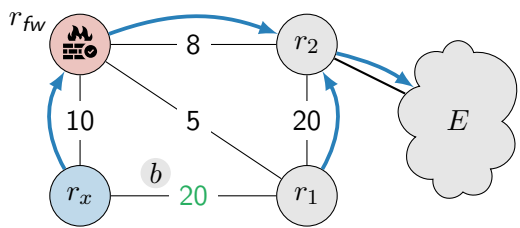
a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Greedy minimization of the cost function



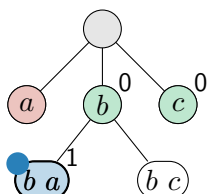


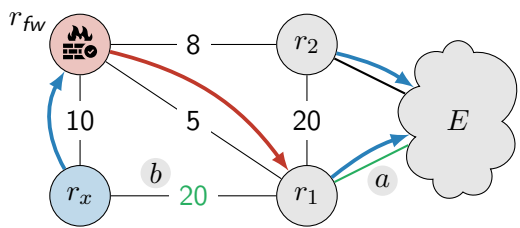
a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Greedy minimization of the cost function



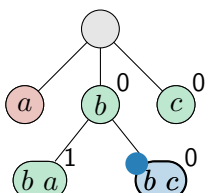


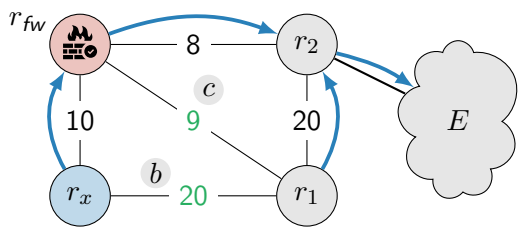
a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Greedy minimization of the cost function



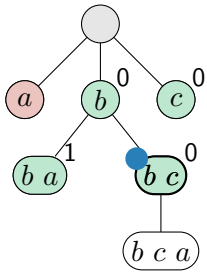


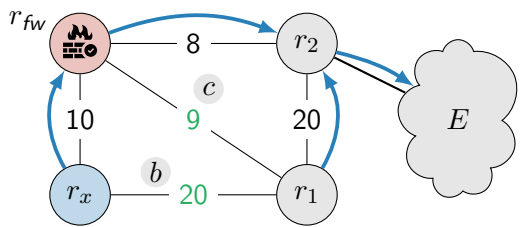
a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Greedy minimization of the cost function



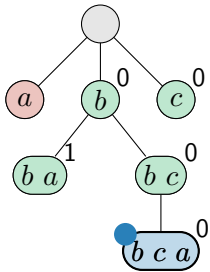


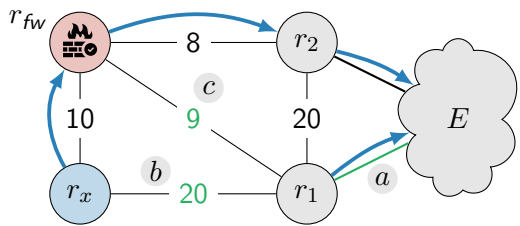
a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Greedy minimization of the cost function



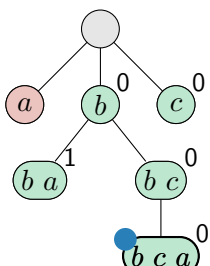


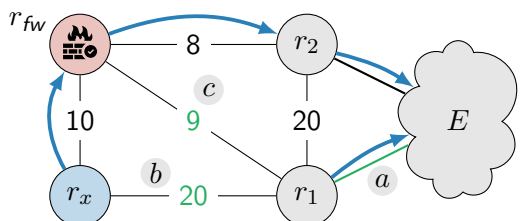
a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

Greedy minimization of the cost function



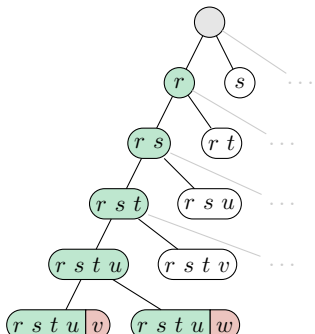


a eBGP session: $E \longrightarrow r_1$

b Link weight r_x — r_1: 10 $\rightsquigarrow$ 20

c Link weight r_{fw} — r_1: 5 $\rightsquigarrow$ 9

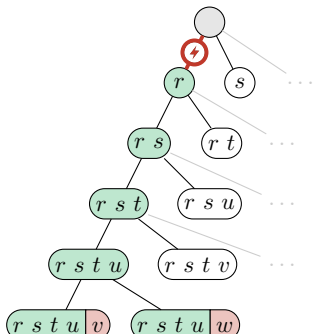
DFS Exploration works well in *most* cases



However: What if we get stuck? Bad decision **early** may cause problems **later**.

→ Actively find the problem!

DFS Exploration works well in *most* cases



However: What if we get stuck? Bad decision **early** may cause problems **later**.

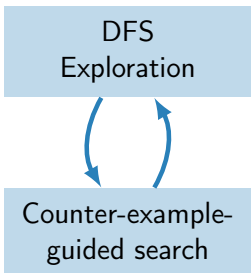
→ Actively find the problem!

1. **Exploration**
   How does Snowcap solve simple scenarios?

2. **Counter-example-guided search**
   How does Snowcap solve more difficult scenarios?

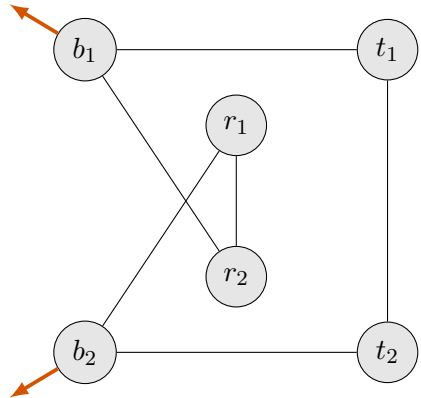
3. **Evaluation**
   How efficient and effective is Snowcap?

Snowcap uses counter-example-guided search to resolve difficult dependencies

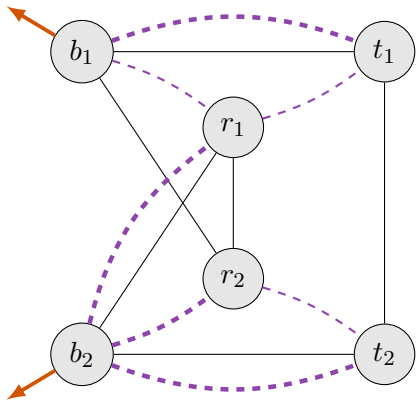


Snowcap ...

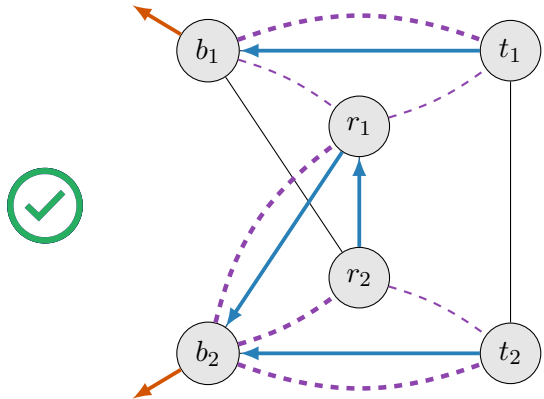
- performs normal exploration **until a dead end**
- follows a **divide-and-conquer** approach

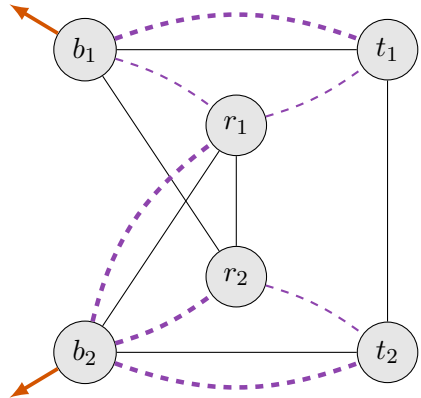

b_1
t_1
r_1
r_2
b_2
t_2

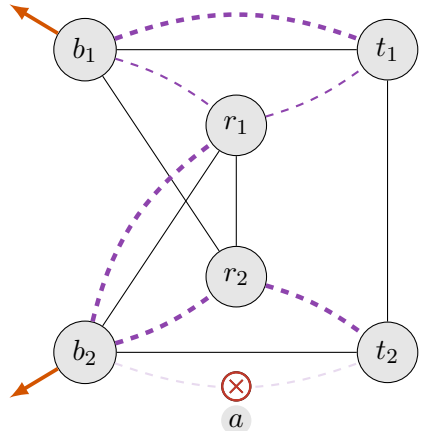
Initial configuration

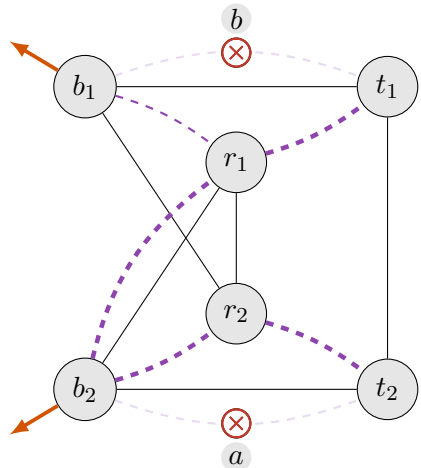


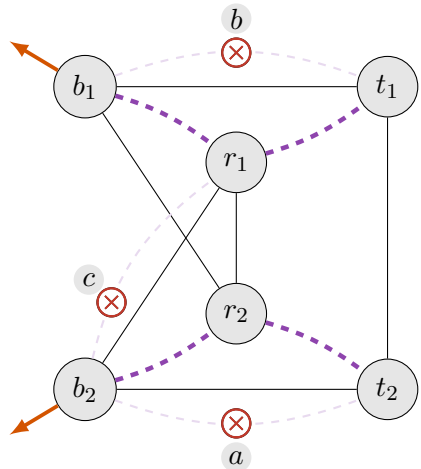
Initial configuration



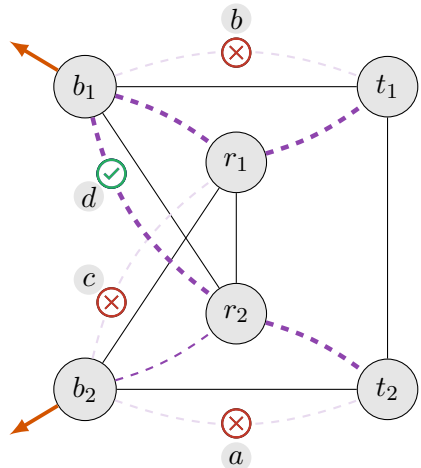

b_1
t_1
r_1
r_2
b_2
t_2


b_1
t_1
r_1
r_2
b_2
t_2
a

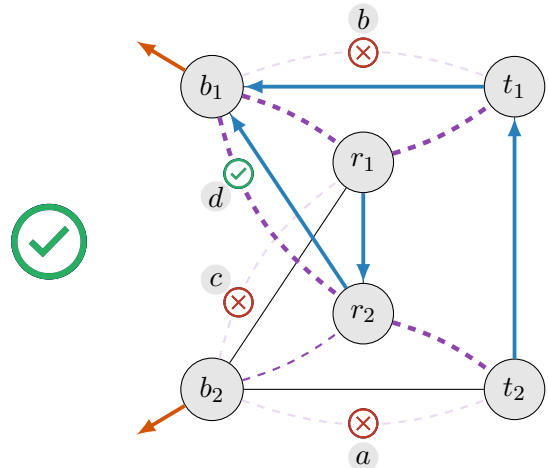

b
b_1
t_1
r_1
r_2
b_2
t_2
a


b
b_1
t_1
r_1
c
r_2
b_2
t_2
a

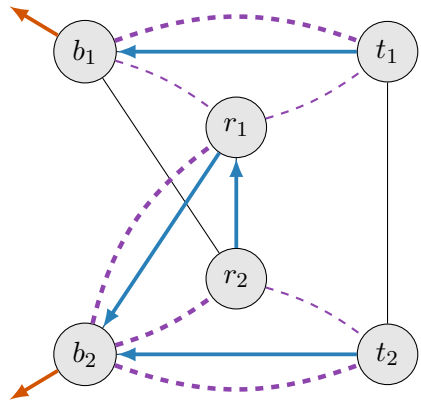
Final configuration



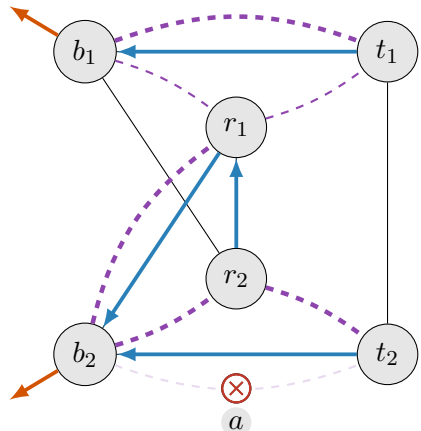
Final configuration



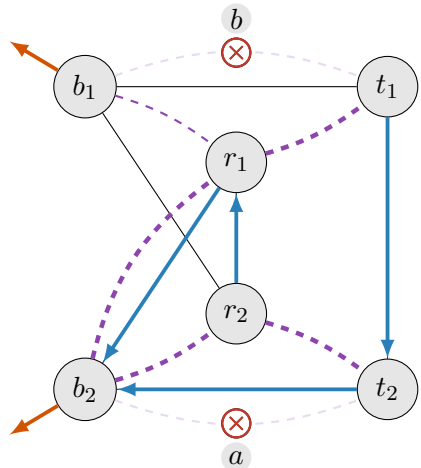
DFS traversal first applies a followed by b



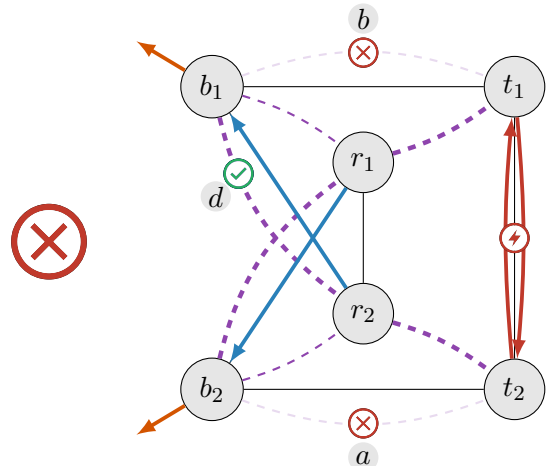
DFS traversal first applies a followed by b



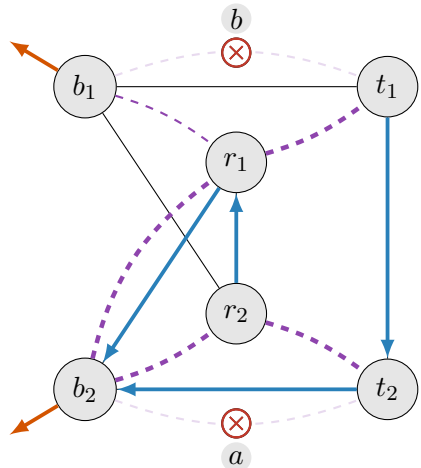
DFS traversal first applies a followed by b



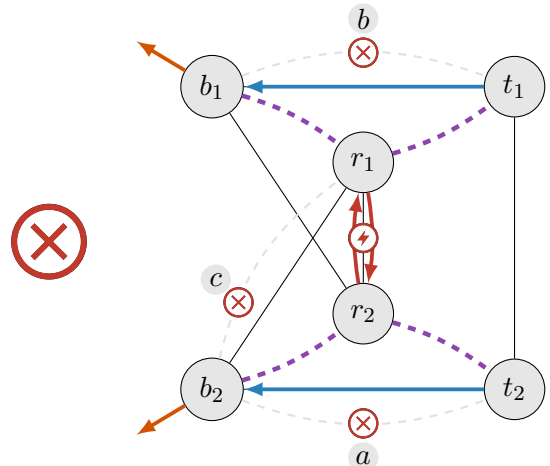
Either c and d will now cause forwarding loops



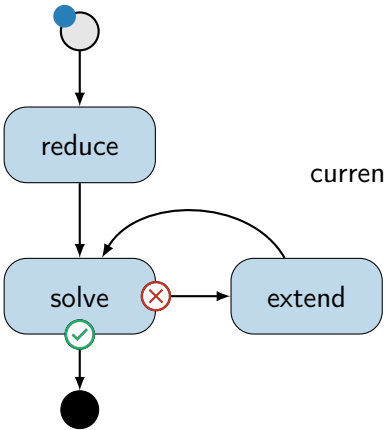
Either c and d will now cause forwarding loops



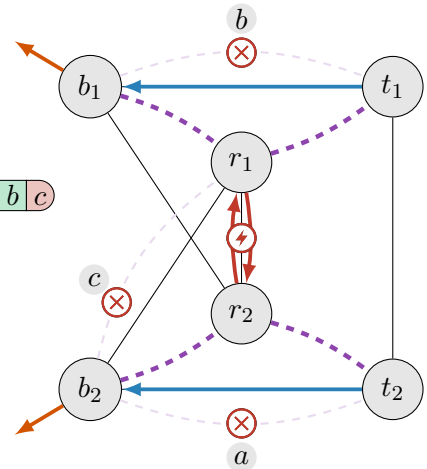
Either c and d will now cause forwarding loops



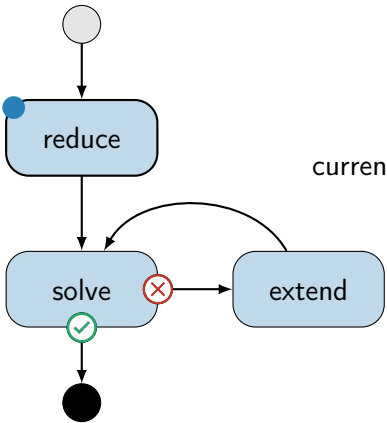
Reduce to minimal reproducing sequence



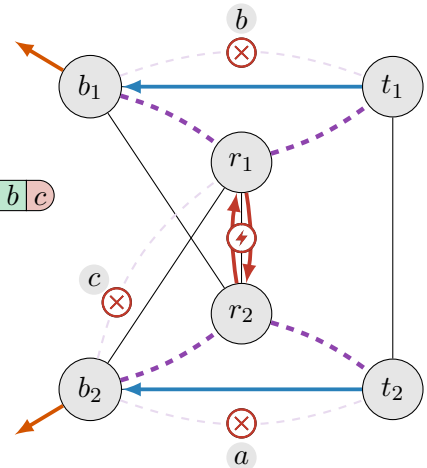
current ordering: $a\ b\ c$



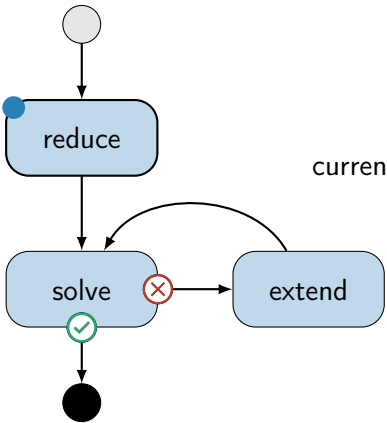
Reduce to minimal reproducing sequence



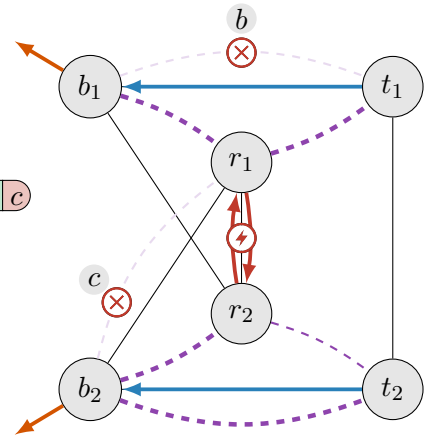
current ordering: $a\ b\ c$



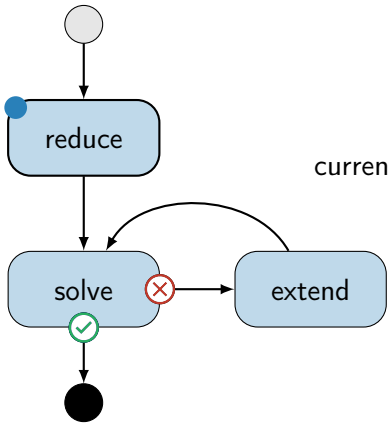
Reduce to minimal reproducing sequence



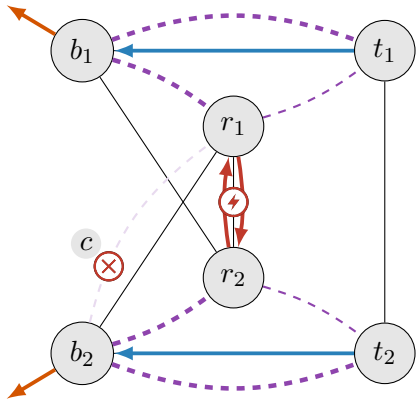
current ordering: b c



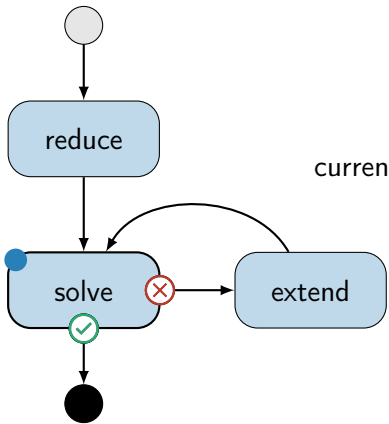
Reduce to minimal reproducing sequence



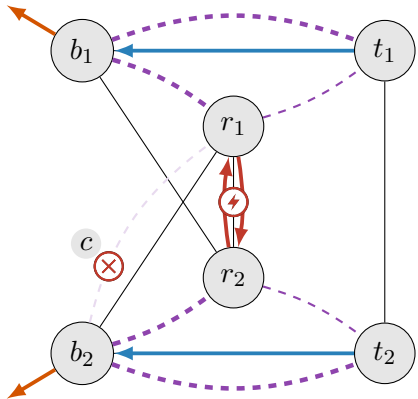
current ordering: c



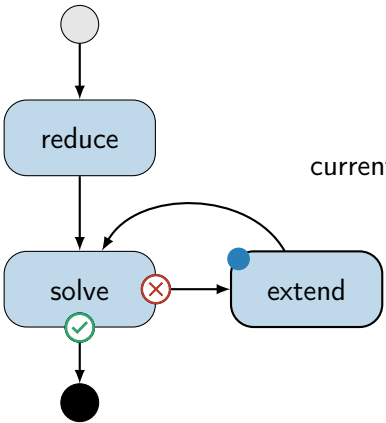
Solve the minimal sequence using the DFS algorithm



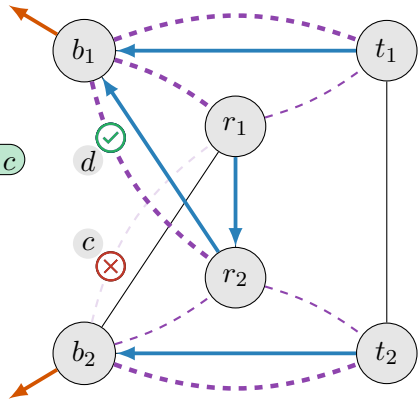
current ordering: c



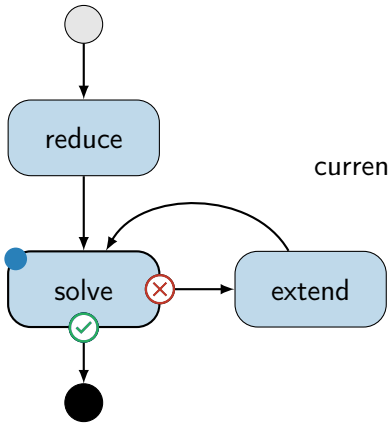
Extend with yet unused commands



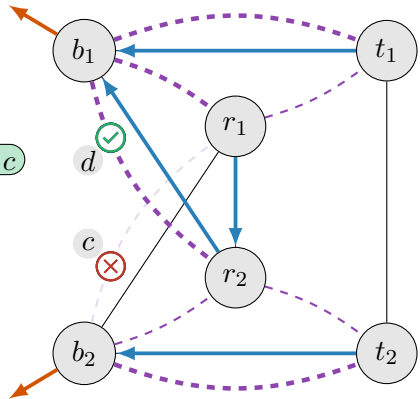
current ordering: $d\ c$



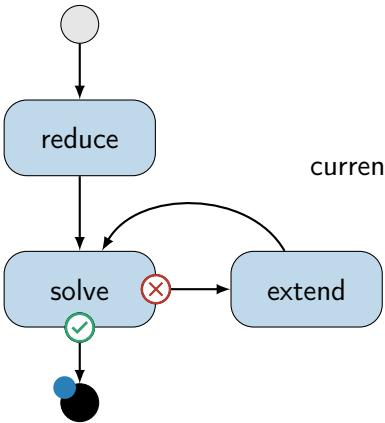
Solve the extended sequence



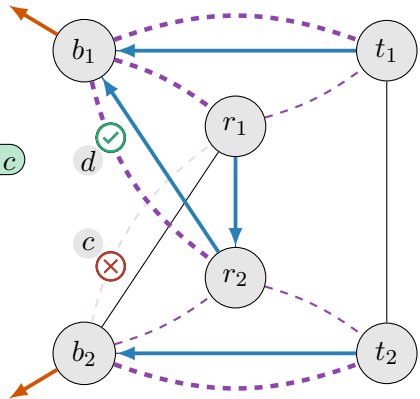
current ordering: $d\ c$



Keep the found solution for future iterations



current ordering: $d\ c$



1. **Exploration**
   How does Snowcap solve simple scenarios?

2. **Counter-example-guided search**
   How does Snowcap solve more difficult scenarios?

3. **Evaluation**
   - How does Snowcap **scale**?
   - How **effective** is Snowcap?

We evaluate Snowcap on a wide range of topologies and migration scenarios

- $\approx$ 80 Topologies from Topology Zoo[6]
- Common migration scenarios[7]
- Random link weights and iBGP topologies.

[6] S. Knight et al. "The Internet Topology Zoo". In: *IEEE JSAC*. 2011.

[7] Gonzalo Gomez Herrero et al. *Network Mergers and Migrations: Junos Design and Implementation*. Vol. 45. John Wiley & Sons, 2011.

Snowcap finds solutions within seconds

Migration from iBGP full-mesh to route-reflection.

$\geq 50\%$ chance to violate reachability	
Random order	$\mathbf{70}\%$
Best practice order	$\mathbf{25}\%$

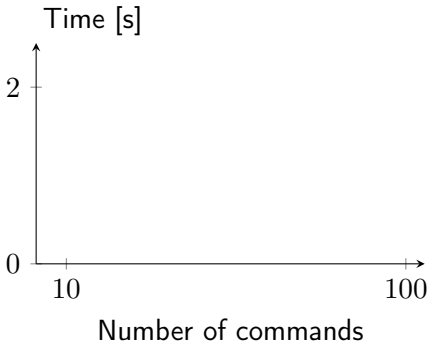
Snowcap finds solutions within seconds

Migration from iBGP full-mesh to route-reflection.

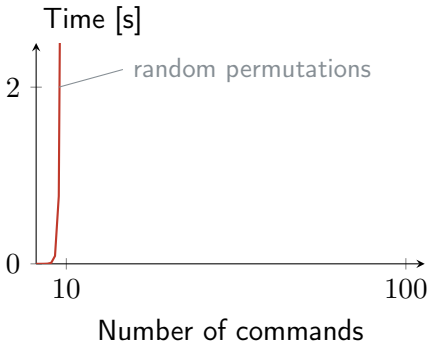
$\geq 50\%$ chance to violate reachability		time
Random order	$\mathbf{70}\%$	
Best practice order	$\mathbf{25}\%$	
Snowcap	$\mathbf{0}\%$	at most $12s^*$

*for 3081 commands on 82 routers.

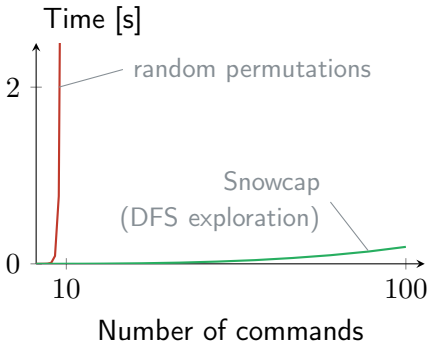
Snowcap's runtime scales very well with increasing complexity



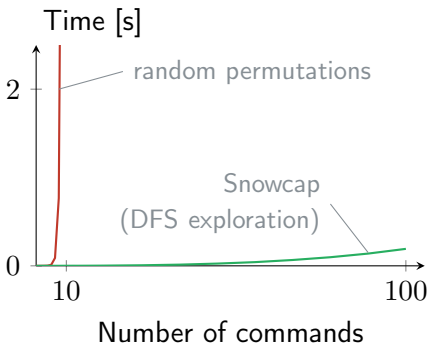
Snowcap's runtime scales very well with increasing complexity

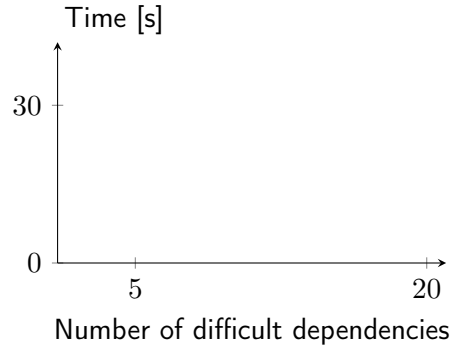


Snowcap's runtime scales very well with increasing complexity

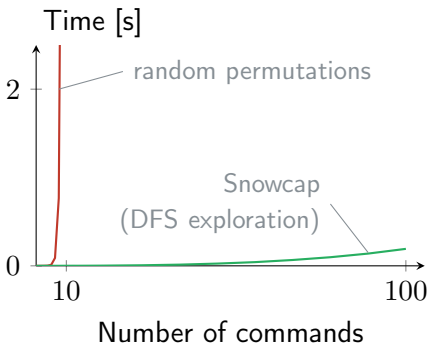


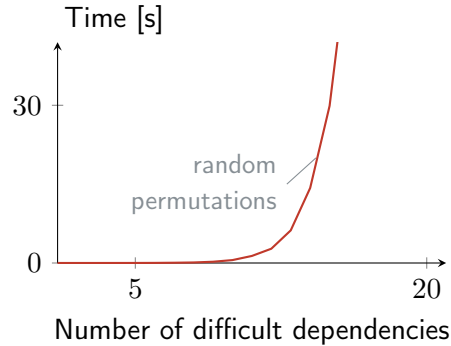
Snowcap's runtime scales very well with increasing complexity



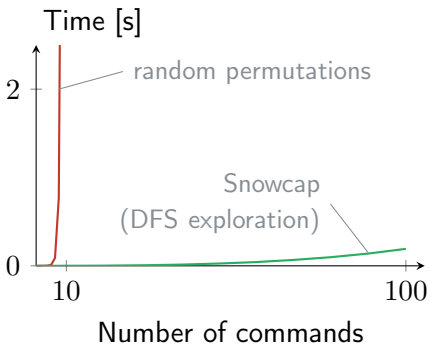


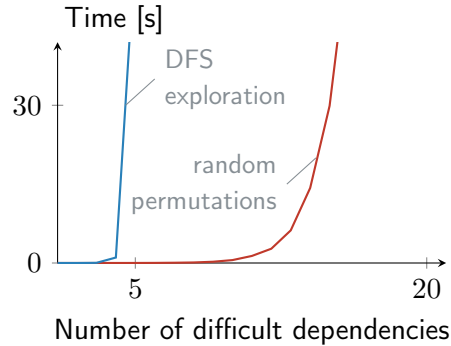
Snowcap's runtime scales very well with increasing complexity



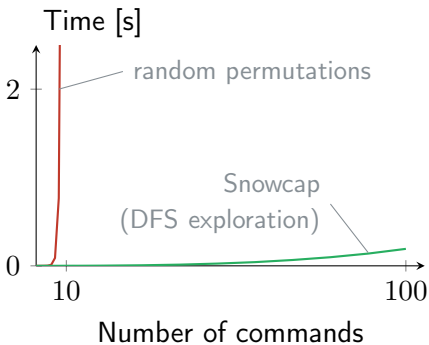


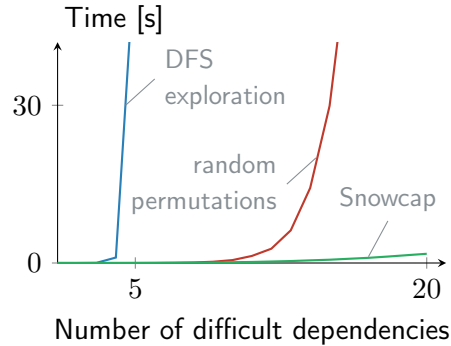
Snowcap's runtime scales very well with increasing complexity



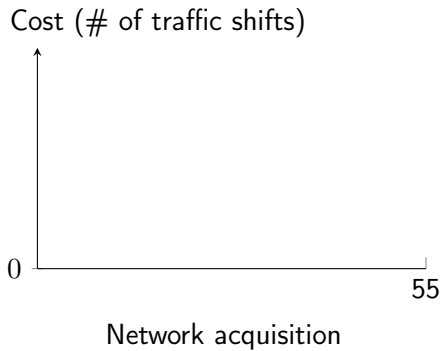


Snowcap's runtime scales very well with increasing complexity

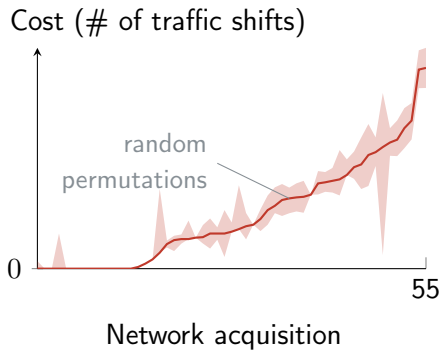




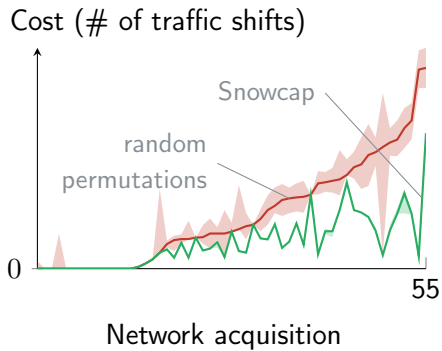
Snowcap is effective at finding good orderings



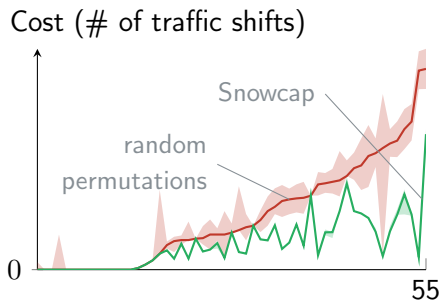
Snowcap is effective at finding good orderings



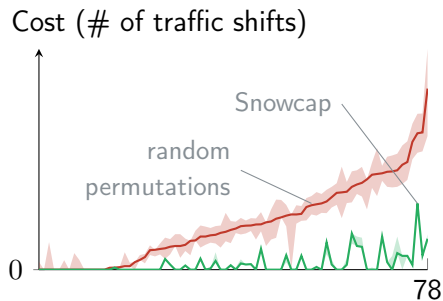
Snowcap is effective at finding good orderings



Snowcap is effective at finding good orderings

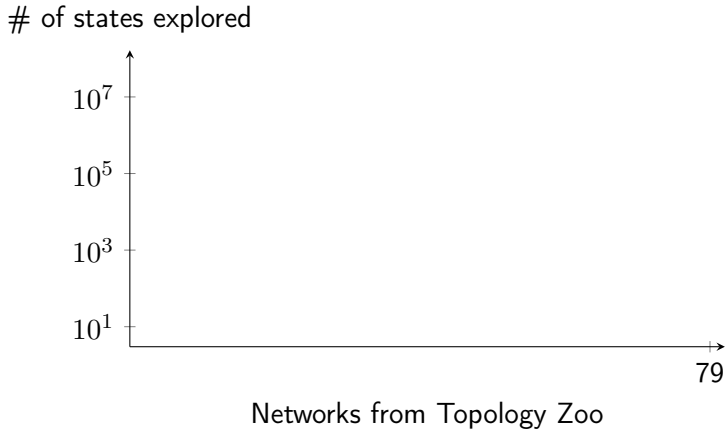


Network acquisition

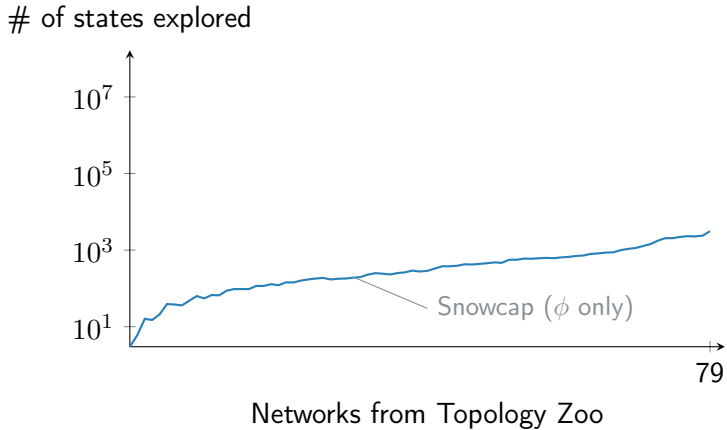


Double all link weights

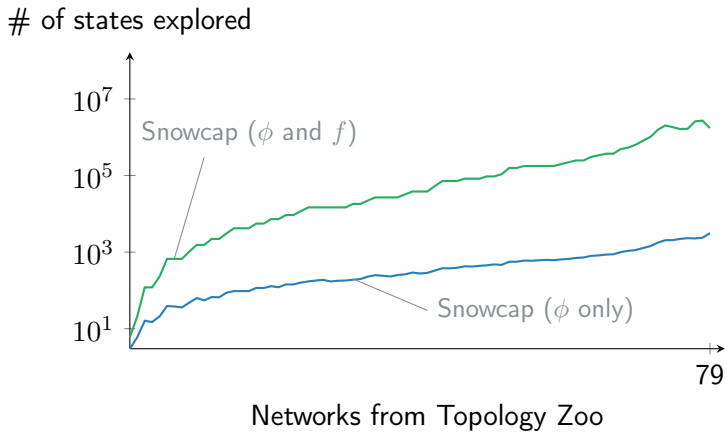
Predictable overhead for Snowcap's greedy optimization



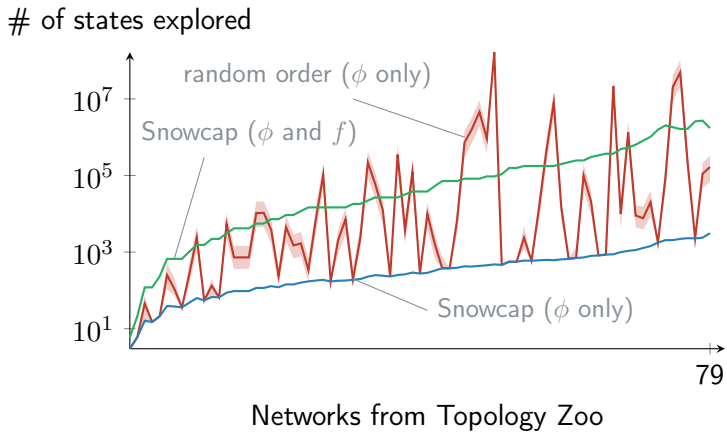
Predictable overhead for Snowcap's greedy optimization



Predictable overhead for Snowcap's greedy optimization



Predictable overhead for Snowcap's greedy optimization



Snowcap guarantees safe network reconfigurations

Snowcap is open-source
https://snowcap.ethz.ch

Snowcap: Synthesizing Network-Wide Configuration Updates

Tibor Schneider Rüdiger Birkner Laurent Vanbever

https://nsg.ee.ethz.ch