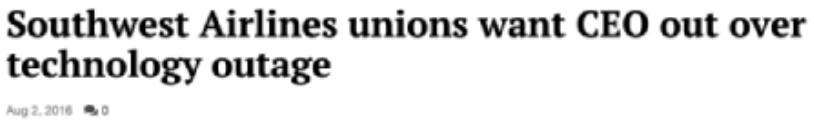
Efficient verification of network fault tolerance via counterexample-guided refinement

Nick Giannarakis [1] Ryan Beckett [2] Ratul Mahajan [3 4] David Walker [1]

[1] Princeton University
[2] Microsoft Research
[3] University of Washington
[4] Intentionet

Network misconfigurations



Configuration challenges

(1) ***Complexity***. Configurations are overly complex.

(2) ***Changing environment***. Peers send new routes.

(3) ***Failures***. Exponential number of behaviors to check.

Configuration challenges

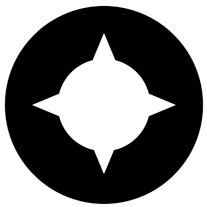
- ✓ ***Complexity***. Configurations are overly complex.
- ✓ ***Changing environment***. Peers send arbitrary routes.
- ✓ ***Failures***. Exponential number of behaviors to check.

MineSweeper [Beckett 2017]

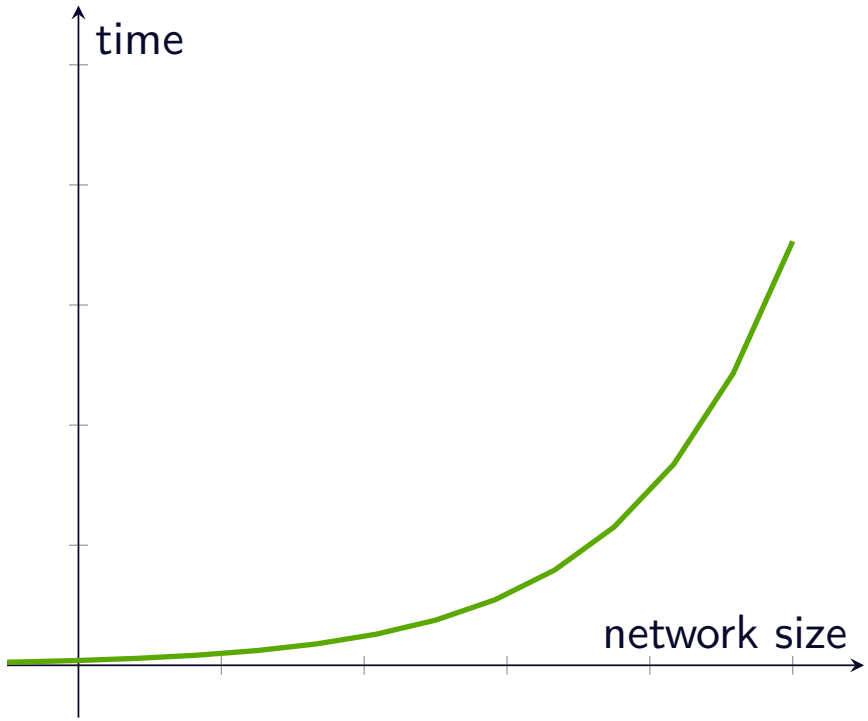
SMT based verifier.

Configuration challenges

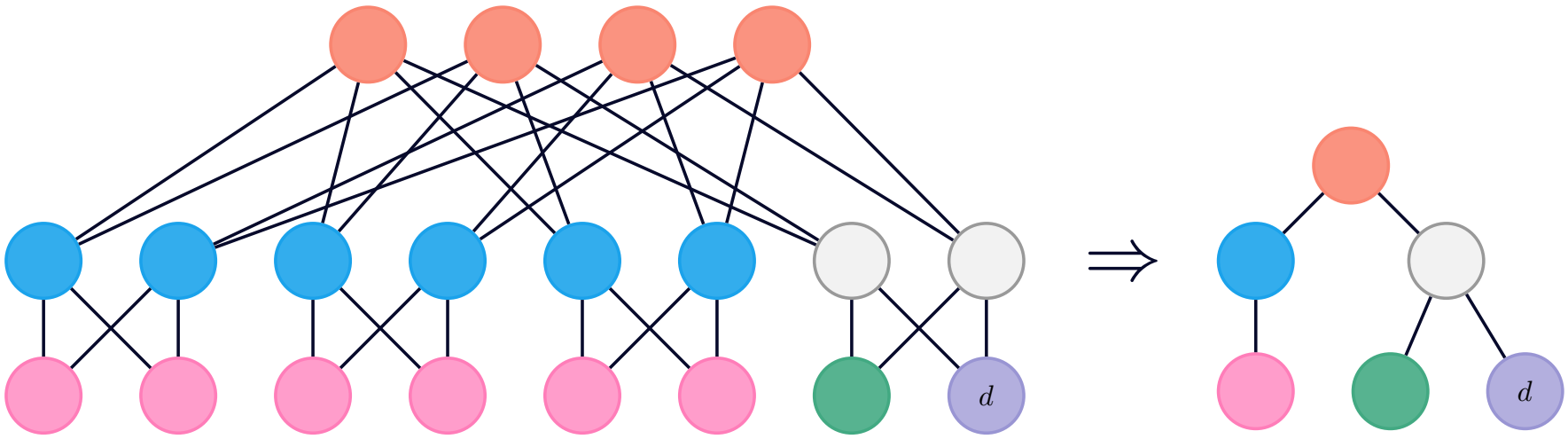
- ✓ ***Complexity***. Configurations are overly complex.
- ✓ ***Changing environment***. Peers send arbitrary routes.
- ✓ ***Failures***. Exponential number of behaviors to check.
- ✗ ***Scale***. Millions of configuration lines on thousands of devices.



MineSweeper [Beckett 2017]
SMT based verifier.



Network compression: Bonsai [Beckett *et al.*, 2018]

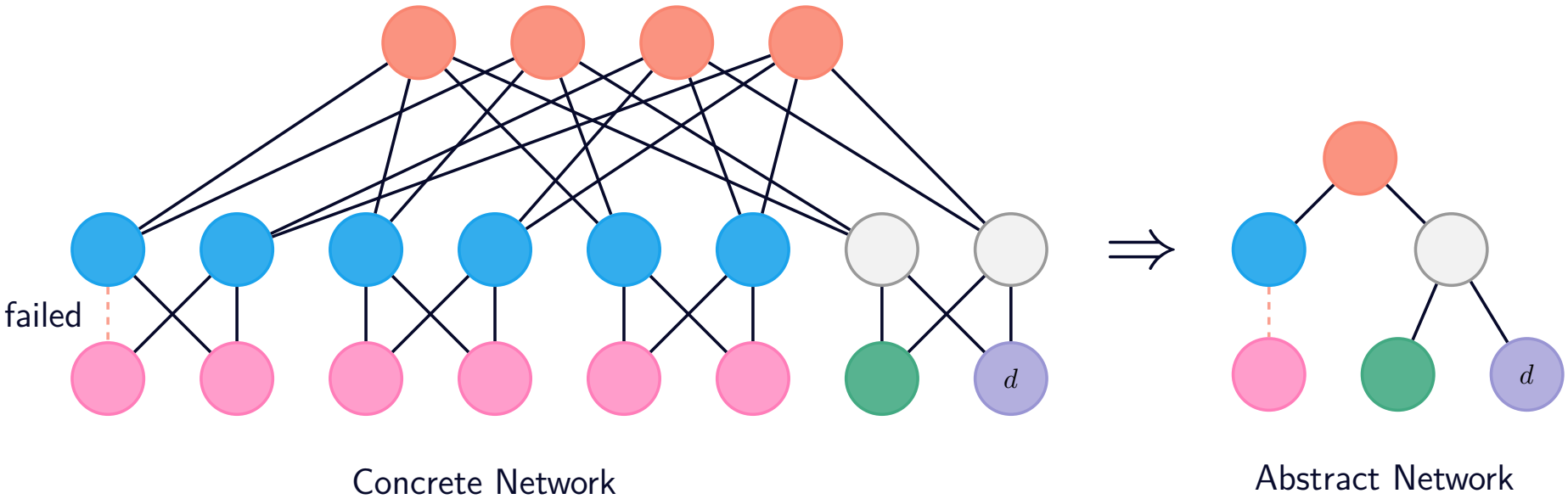


Concrete Network

Abstract Network

- Exploit topology/policy *symmetries*.
- Concrete nodes route "in the same way" as their abstraction.

Network compression: Bonsai [Beckett *et al.*, 2018]



Concrete Network

Abstract Network

- Exploit topology/policy *symmetries*.
- Concrete nodes route "in the same way" as their abstraction.
- But: does not preserve fault tolerance properties!

Our contribution

Is compression possible in the presence of failures?

Our contribution

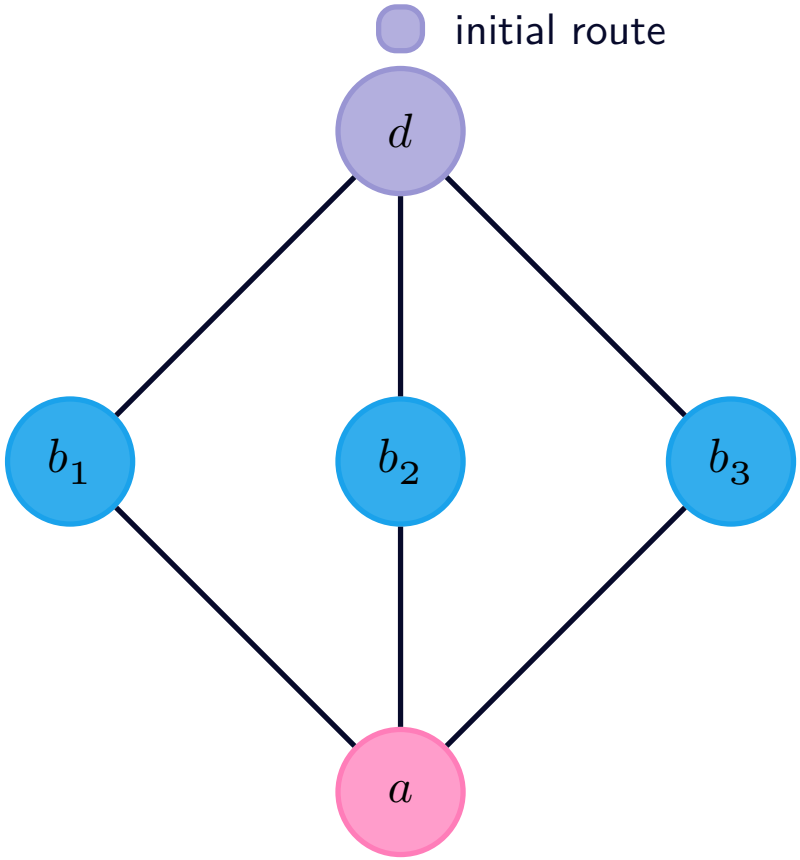
Is compression possible in the presence of failures?

Yes! In this talk:

- A network compression theory compatible with failures.
- ***Origami***, a tool that combines graph algorithms and SMT reasoning to compress a network and verify reachability properties in the presence of link failures.

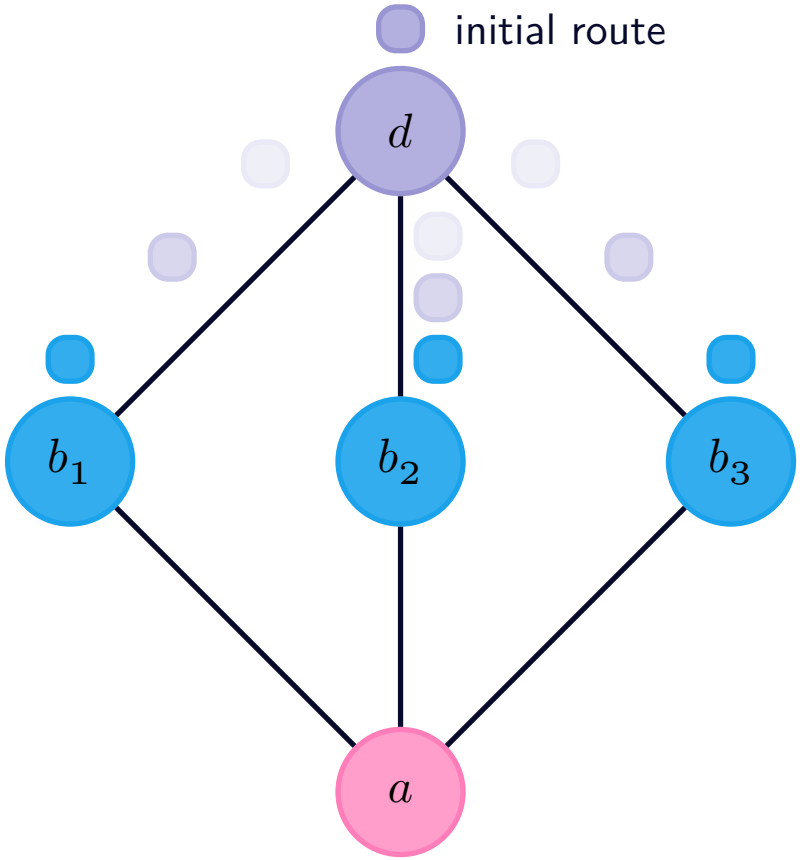
Network Compression Theory

The routing problem



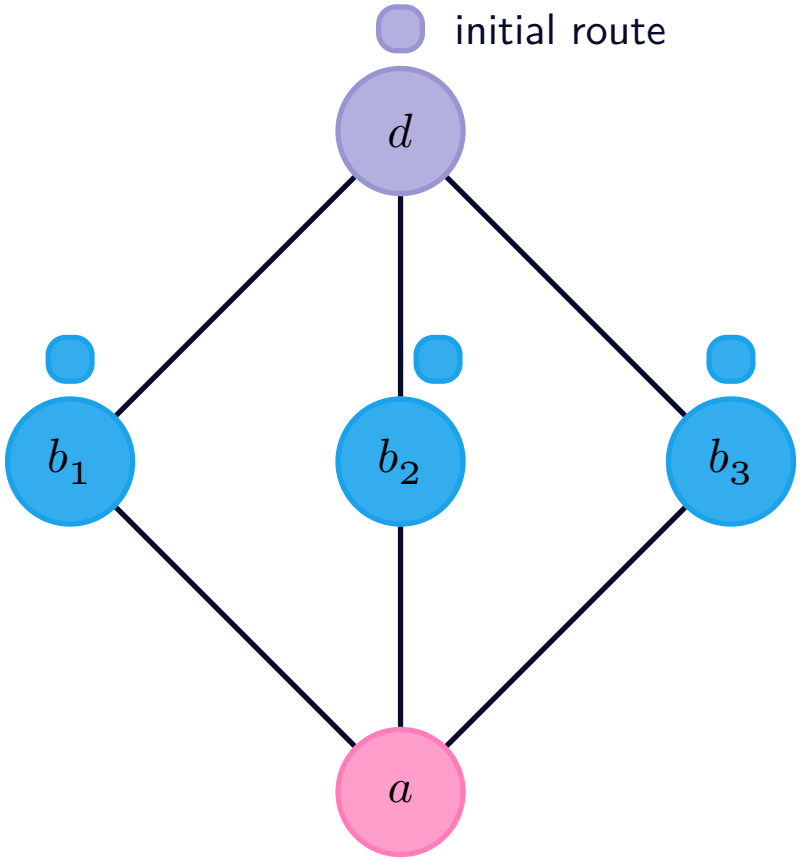
Formal model: Stable paths [Griffin *et al.*, 2002], routing algebras [Sobrinho, 2005].

The routing problem



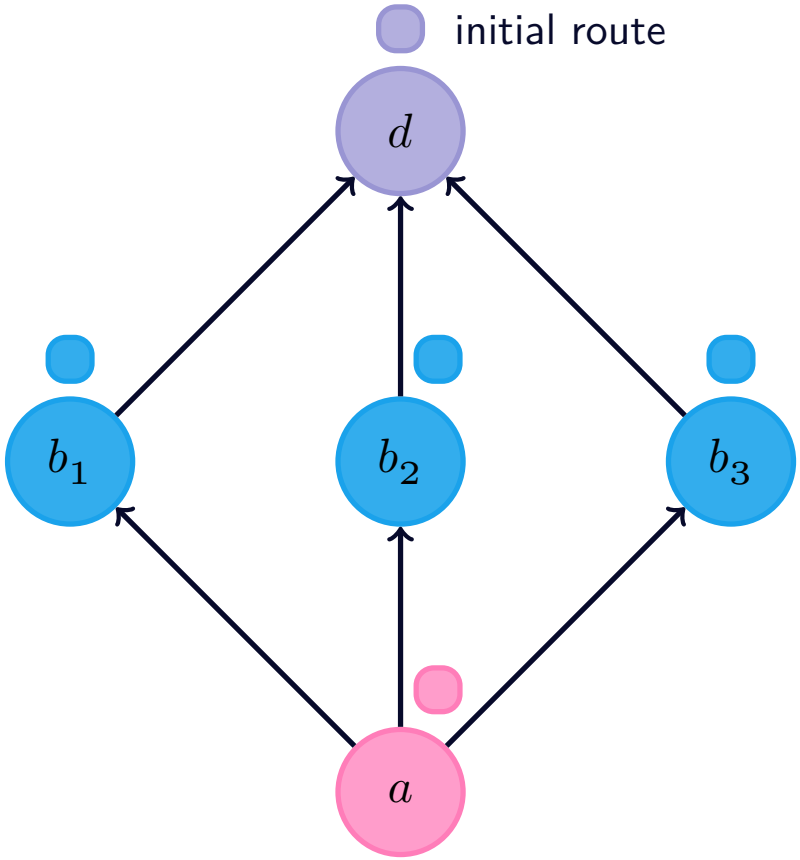
Formal model: Stable paths [Griffin *et al.*, 2002], routing algebras [Sobrinho, 2005].

The routing problem



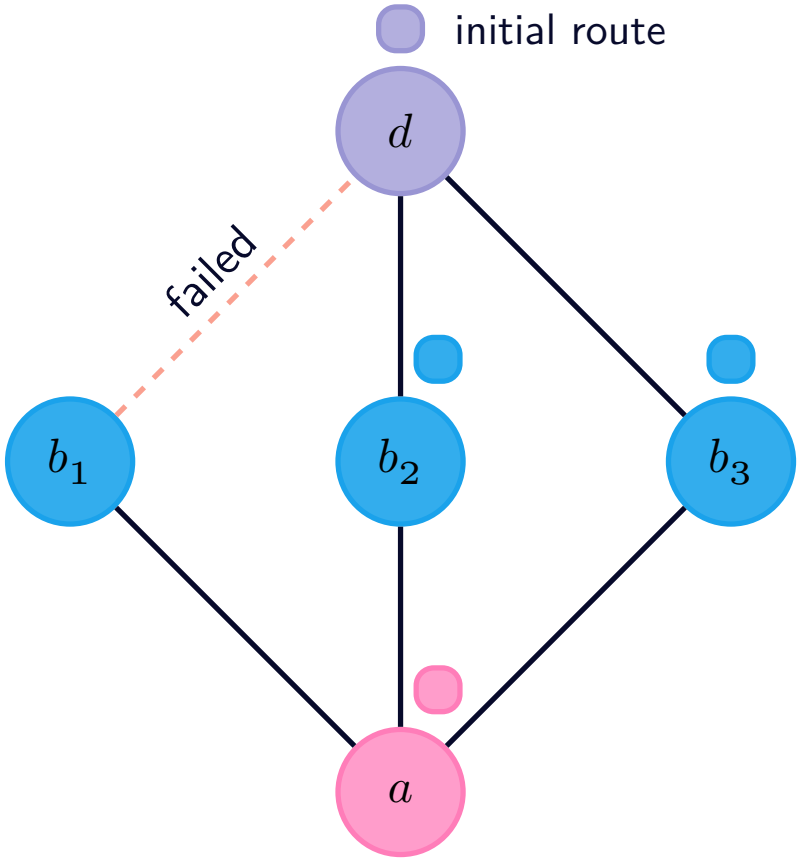
Formal model: Stable paths [Griffin *et al.*, 2002], routing algebras [Sobrinho, 2005].

The routing problem



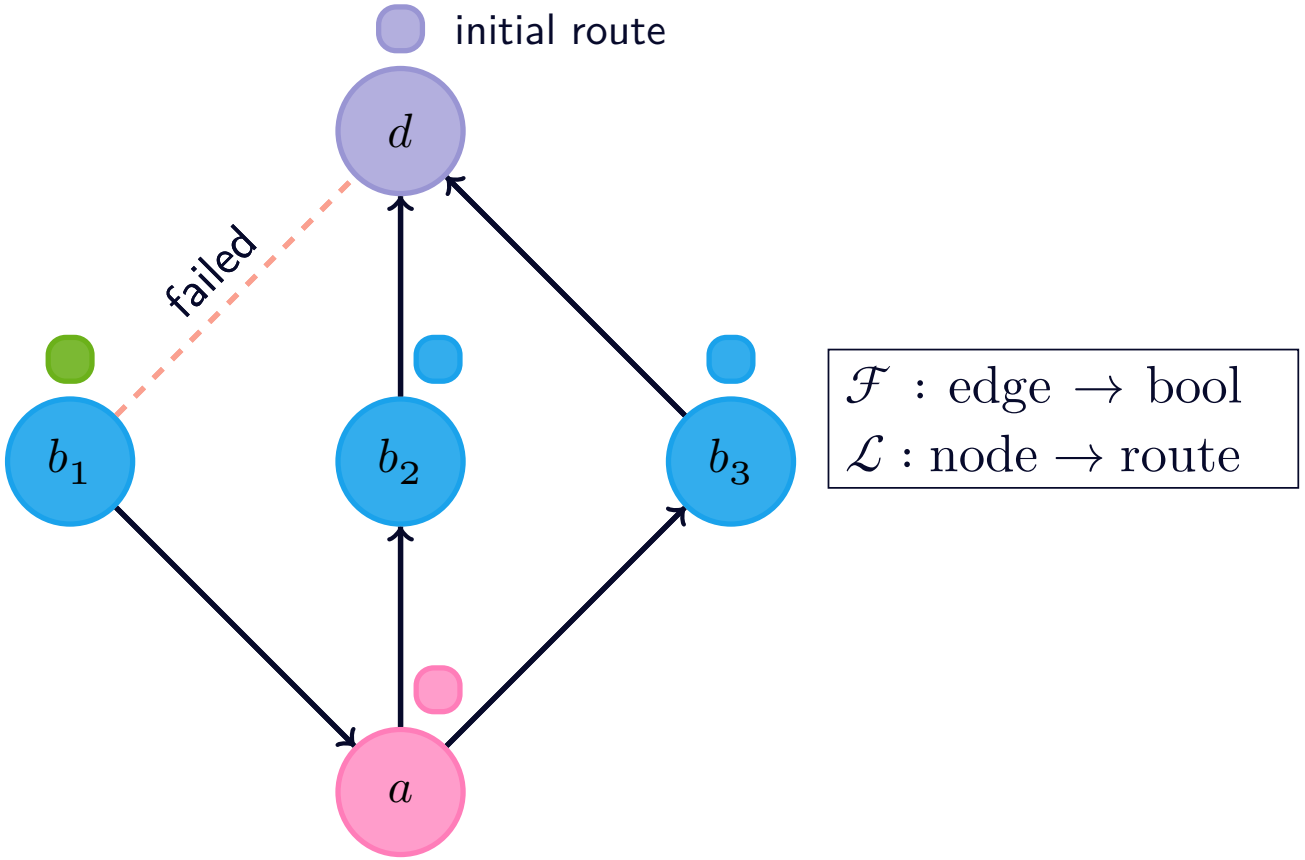
Formal model: Stable paths [Griffin *et al.*, 2002], routing algebras [Sobrinho, 2005].

The routing problem



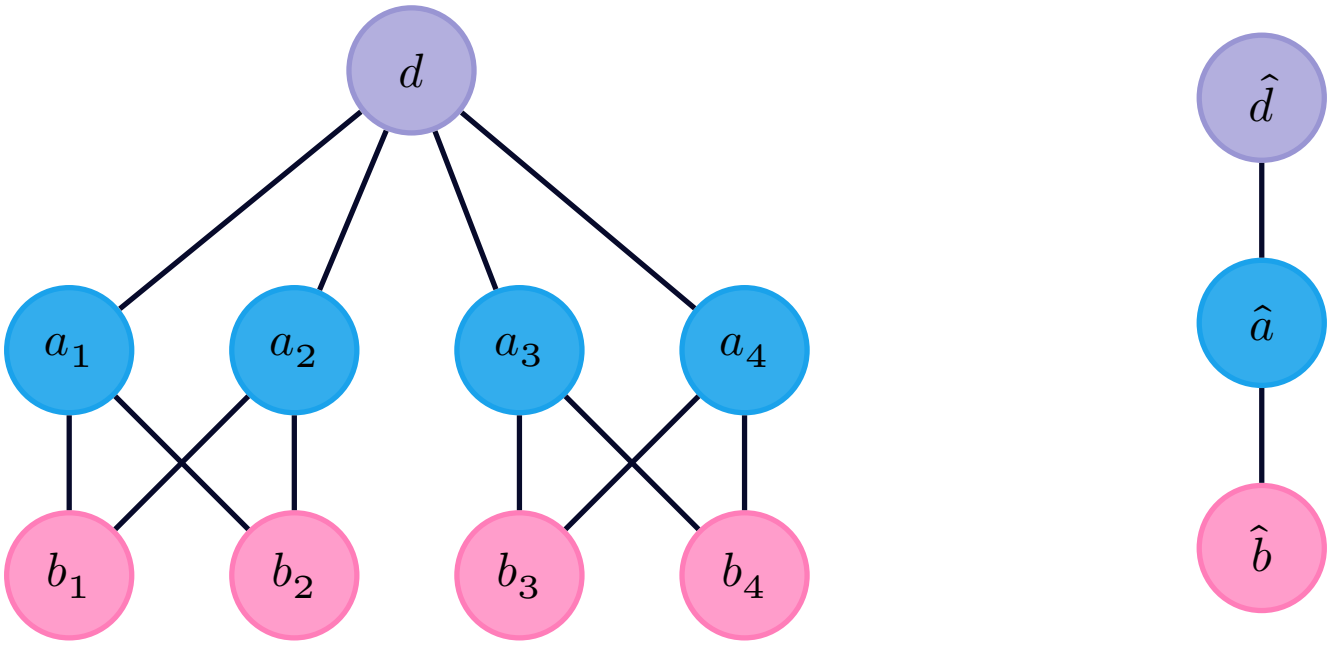
Formal model: Stable paths [Griffin *et al.*, 2002], routing algebras [Sobrinho, 2005].

The routing problem



Formal model: Stable paths [Griffin *et al.*, 2002], routing algebras [Sobrinho, 2005].

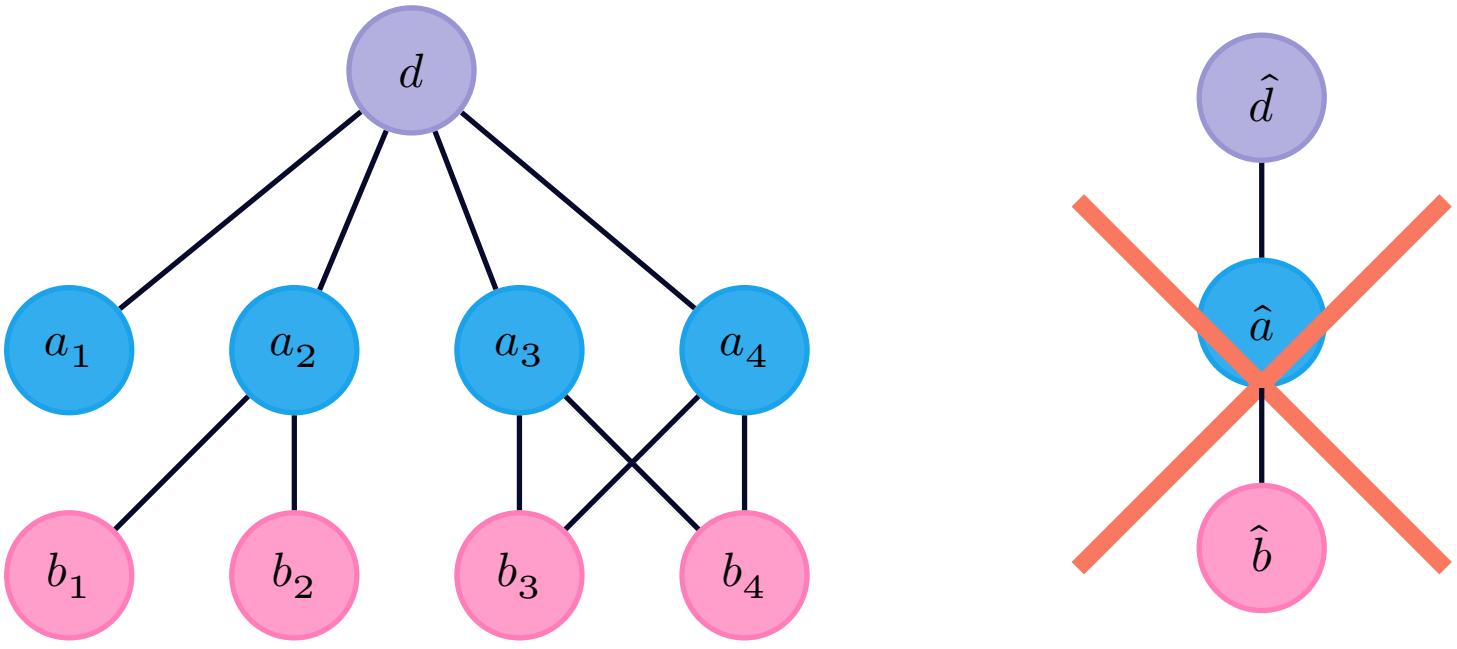
Topological symmetries: $\forall\exists$-abstraction



- Concrete and abstract networks have similar connectivity.
- Example:

  blue abstract node has an edge to pink abstract node iff ***all***
  blue concrete nodes have an edge to ***some*** pink concrete node.

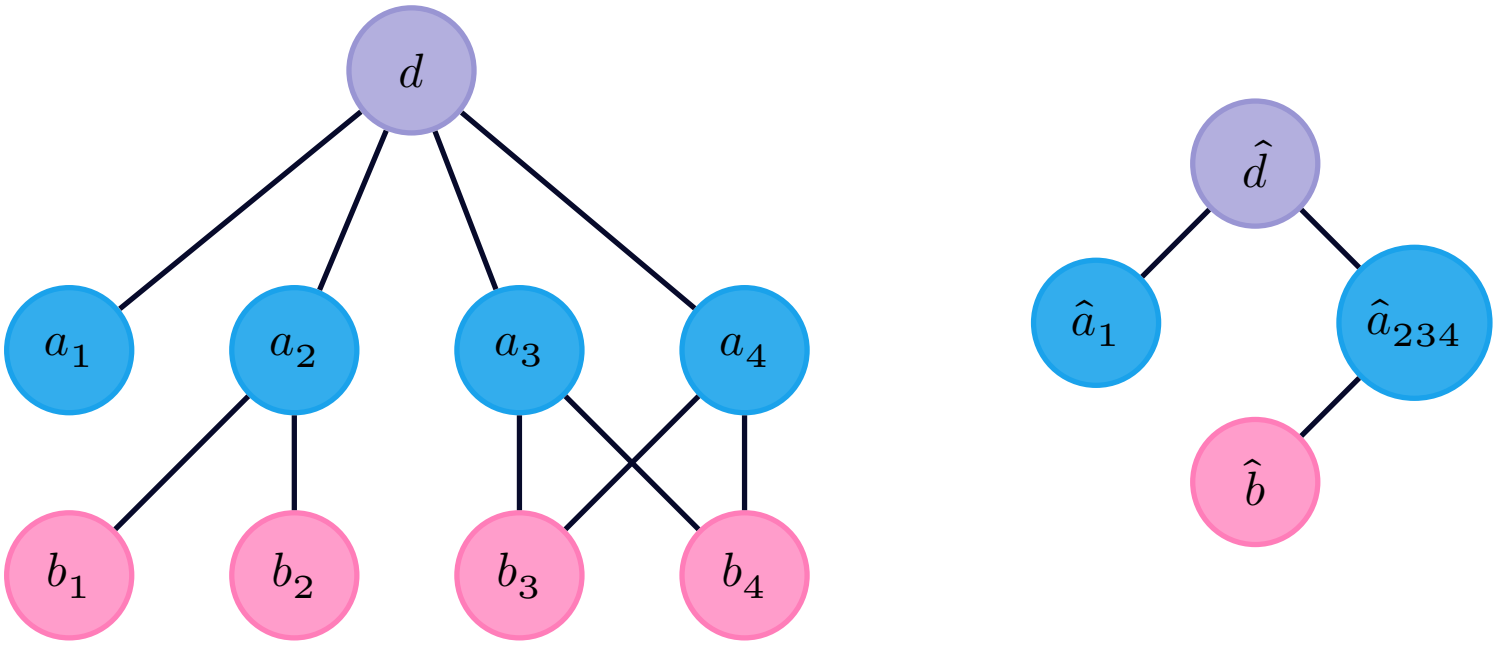
Topological symmetries: $\forall\exists$-abstraction



- Concrete and abstract networks have similar connectivity.
- Example:

  blue abstract node has an edge to pink abstract node iff ***all***
  blue concrete nodes have an edge to ***some*** pink concrete node.

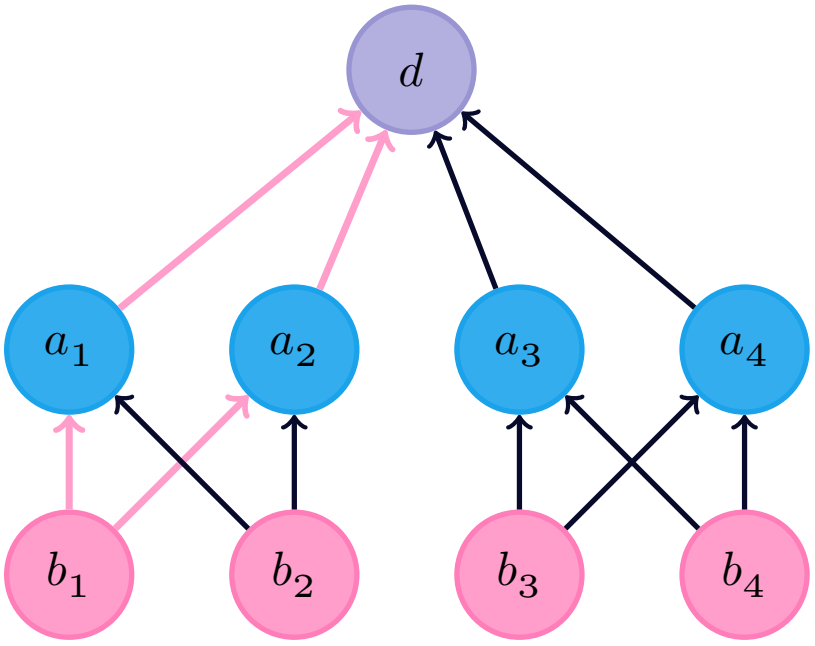
Topological symmetries: $\forall\exists$-abstraction

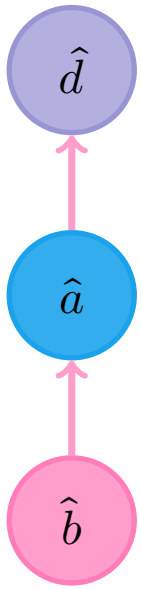


- Concrete and abstract networks have similar connectivity.
- Example:

  blue abstract node has an edge to pink abstract node iff ***all***
  blue concrete nodes have an edge to ***some*** pink concrete node.

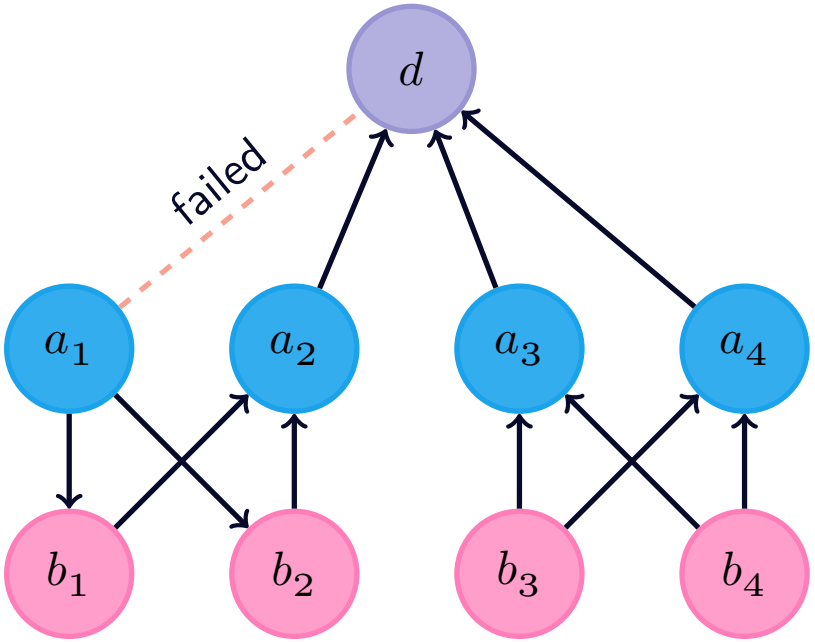
Challenges with link failures I

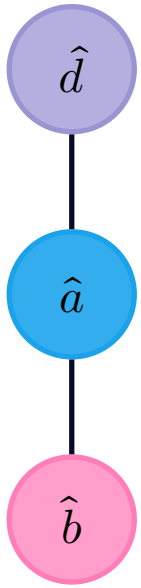




- Concrete pink nodes have 2 disjoint paths, their abstraction has only 1.

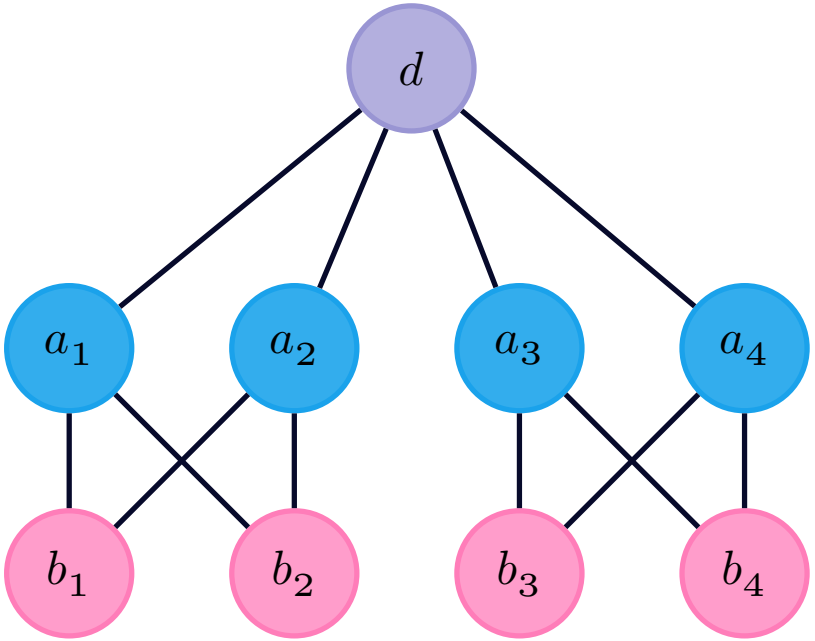
Challenges with link failures II

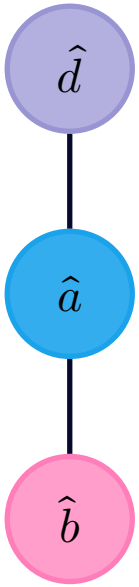




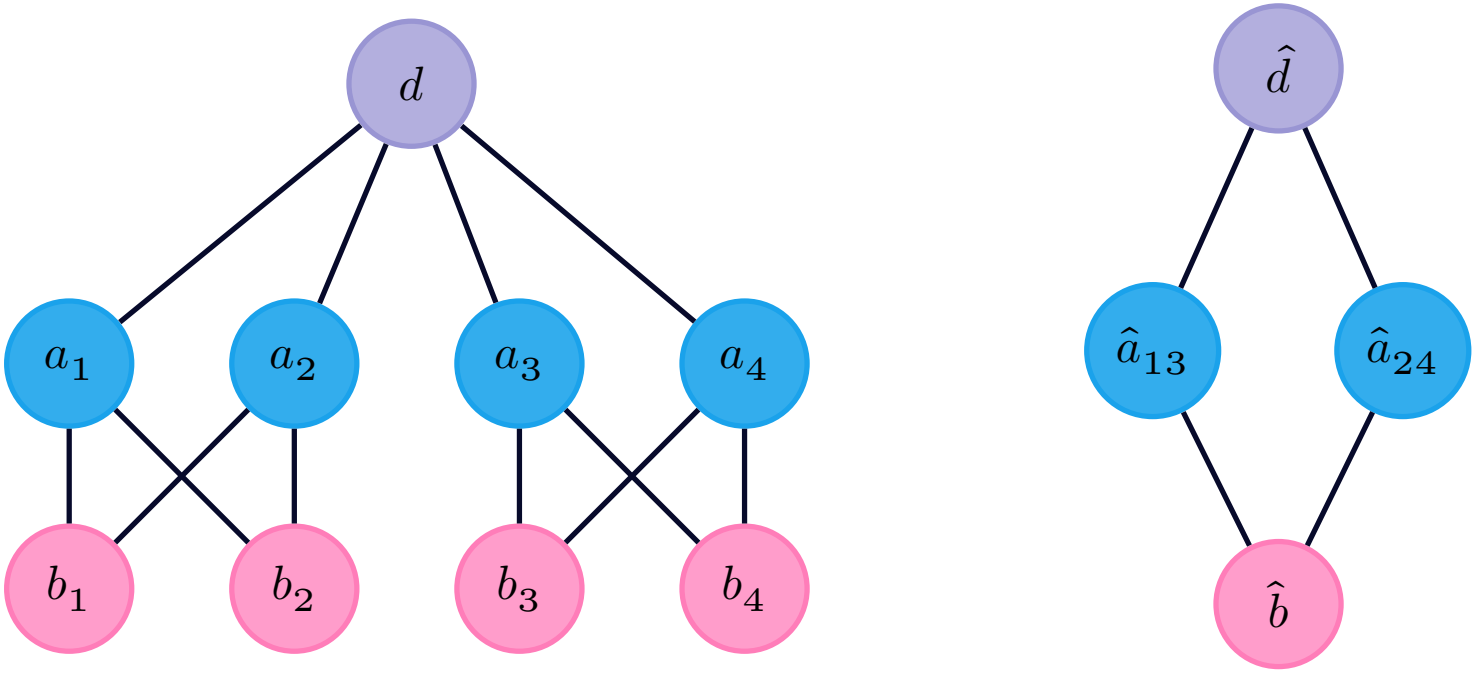
- a_1 no longer has similar routing behavior with a_2, a_3 and a_4.
- $\hat{a}$ does not capture both behaviors.

Plausible abstractions



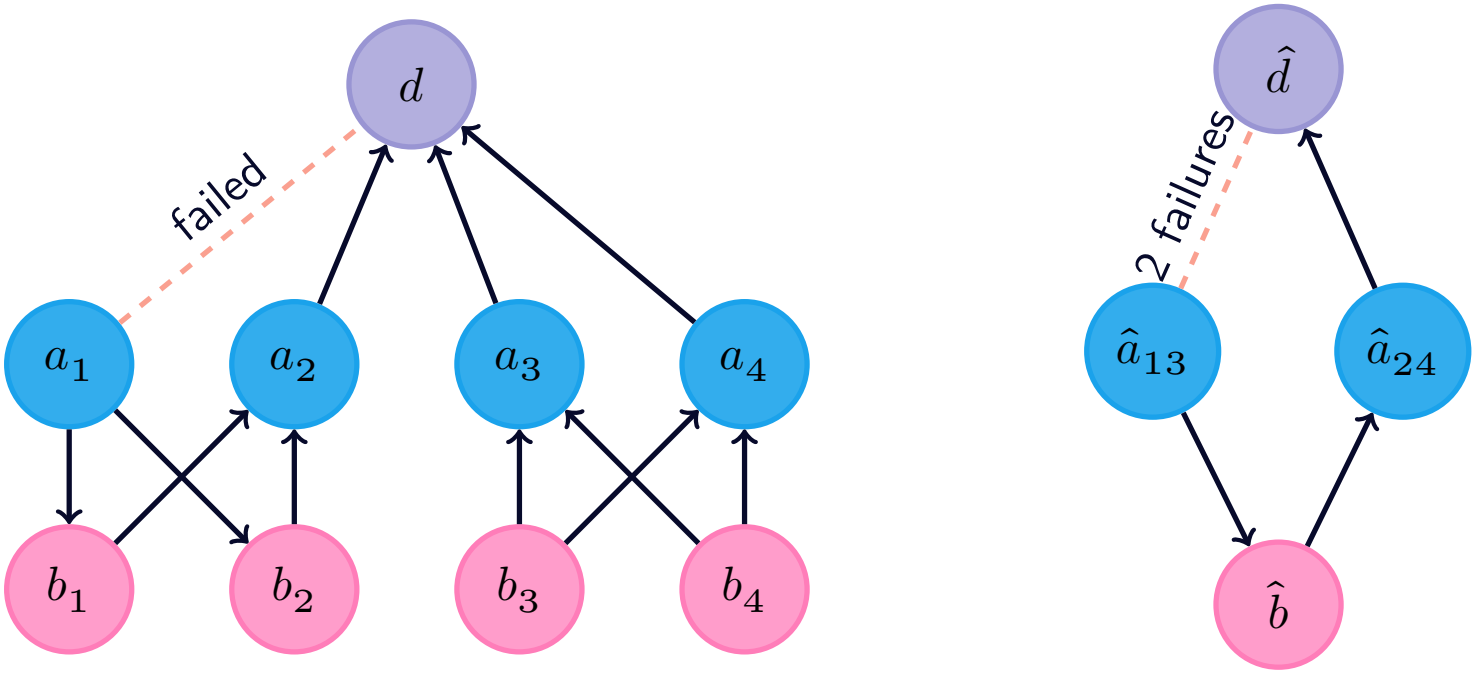


Plausible abstractions



- *Plausible abstraction*: Nodes have two disjoint paths!
- A necessary (but not sufficient) condition for 1-fault tolerance.

Approximating concrete networks



If a node has a route to the destination with k failures then it has a route that is **at least as good** *with k' failures (k' < k).*

- Abstract network over-approximates link failures.
- Approximation is key to achieving compression.

Theory of approximation

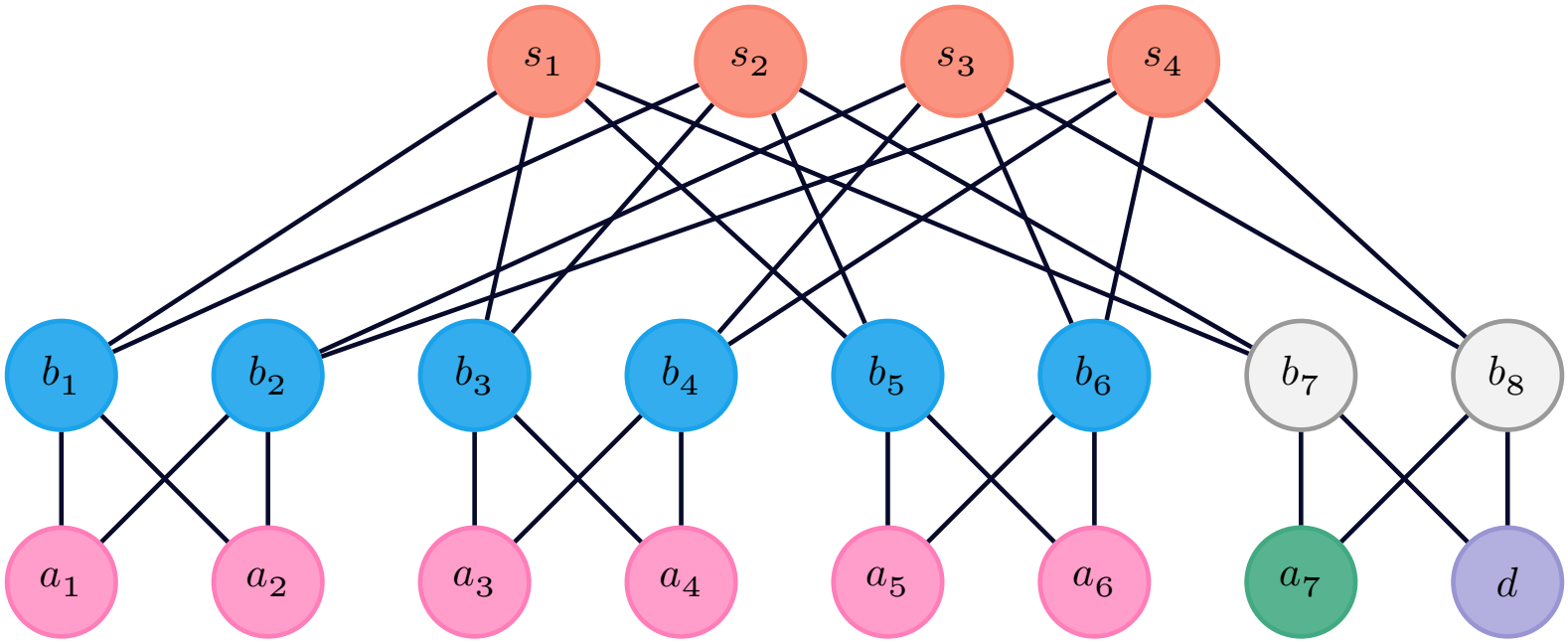
Label approximation theorem

Given a network and its effective abstraction f, for any solution $(\mathcal{L}, \mathcal{F})$ of the concrete network there exists a solution $(\widehat{\mathcal{L}}, \widehat{\mathcal{F}})$ of the abstract network, such that $\mathcal{L}(u) \preceq \widehat{\mathcal{L}}(f(u))$.

- Holds for networks whose policy is *monotonic* and *isotonic*.
  - Monotonic: $\mathrm{route} \prec f(\mathrm{route})$
  - Isotonic: $\mathrm{route}_1 \prec \mathrm{route}_2 \Rightarrow f(\mathrm{route}_1) \prec f(\mathrm{route}_2)$
- Reachability in abstraction implies reachability in the concrete.

Abstraction Algorithm

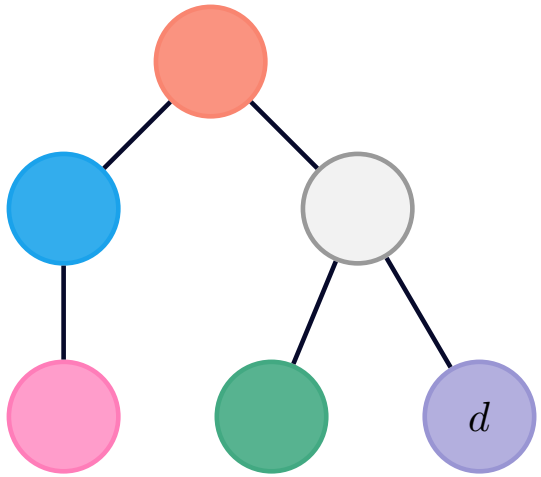
Abstract + verify



- pink nodes do not announce routes to blue nodes.

(1) Can blue and pink nodes reach the destination when there is 1 link failure?

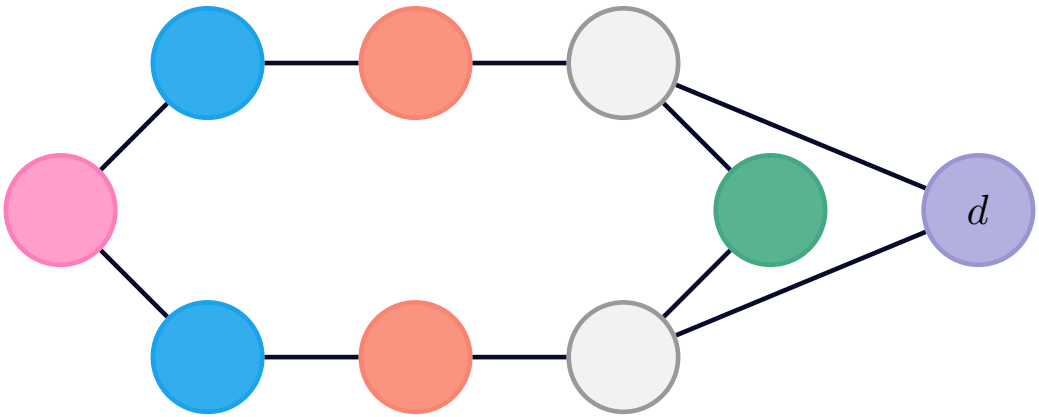
Abstract + verify



(1) Can blue and pink nodes reach the destination when there is 1 link failure?

(2) Start from the smallest abstraction.

Abstract + verify

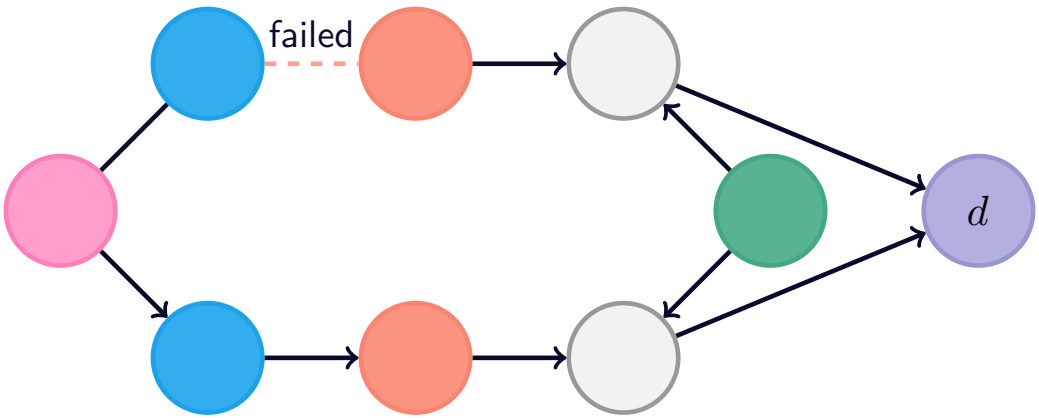


(1) Can blue and pink nodes reach the destination when there is 1 link failure?

(2) Start from the smallest abstraction.

(3) `REFINE` to obtain a plausible abstraction:
$|\mathrm{mincut}(\mathsf{Graph},\mathsf{blue})| > 1$ and $|\mathrm{mincut}(\mathsf{Graph},\mathsf{pink})| > 1$.

Abstract + verify



(1) Can blue and pink nodes reach the destination when there is 1 link failure?

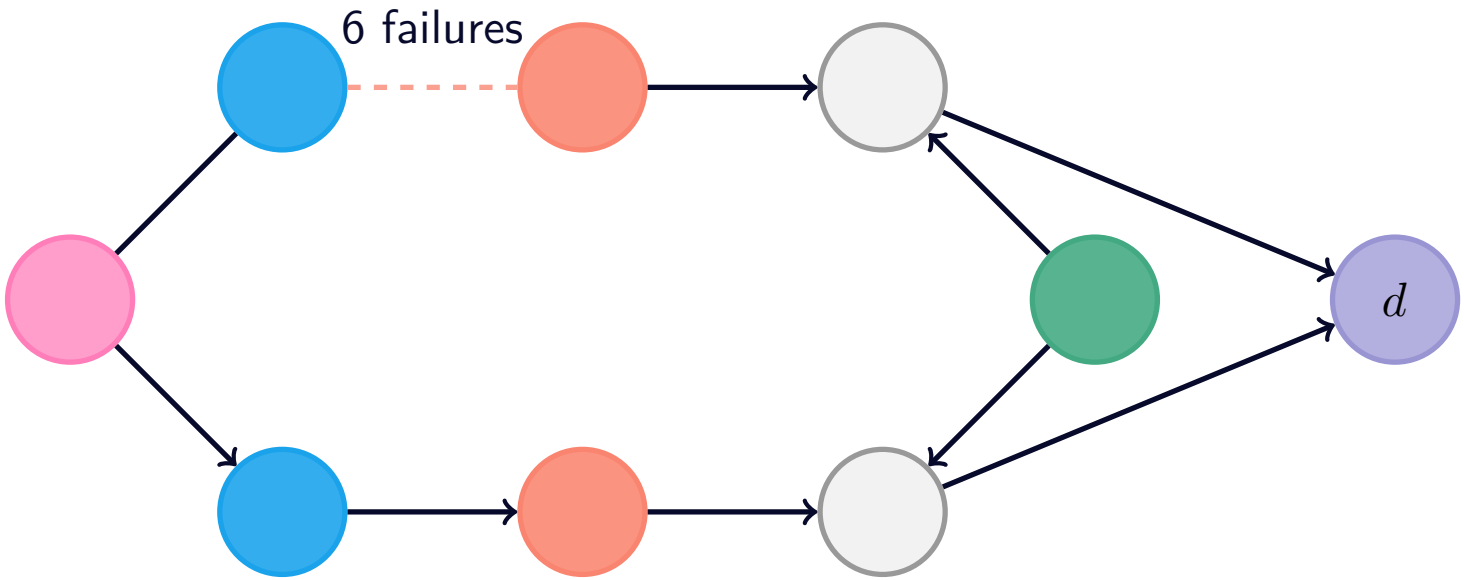
(2) Start from the smallest abstraction.

(3) `REFINE` to obtain a plausible abstraction:
$|\mathrm{mincut}(\text{Graph},\text{blue})| > 1$ and $|\mathrm{mincut}(\text{Graph},\text{pink})| > 1$.

(4) Run the verification procedure.

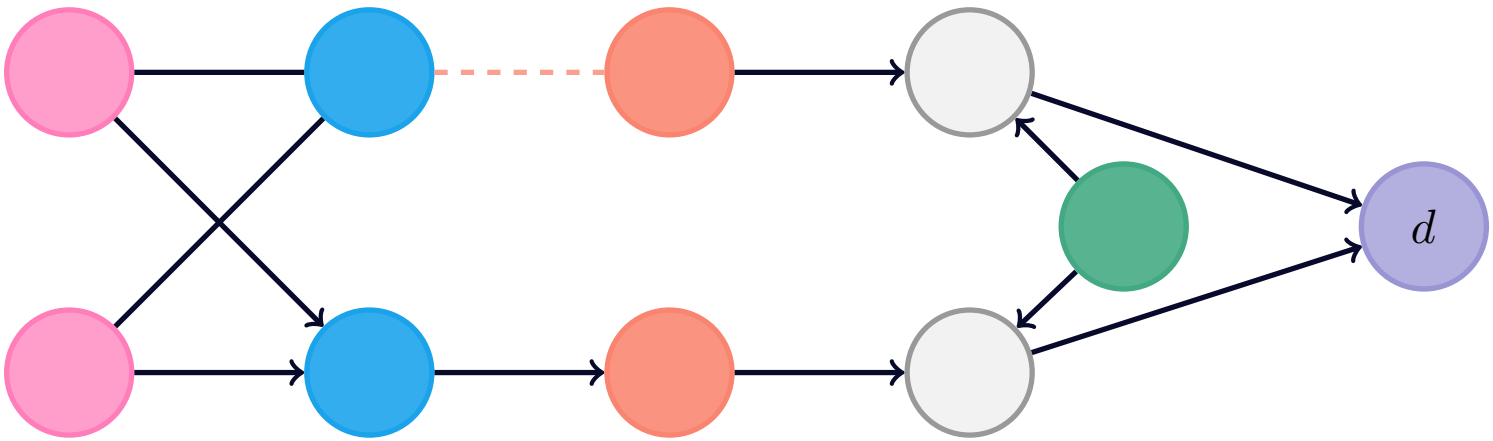
- blue node cannot reach the destination!

Counterexample-guided refinement



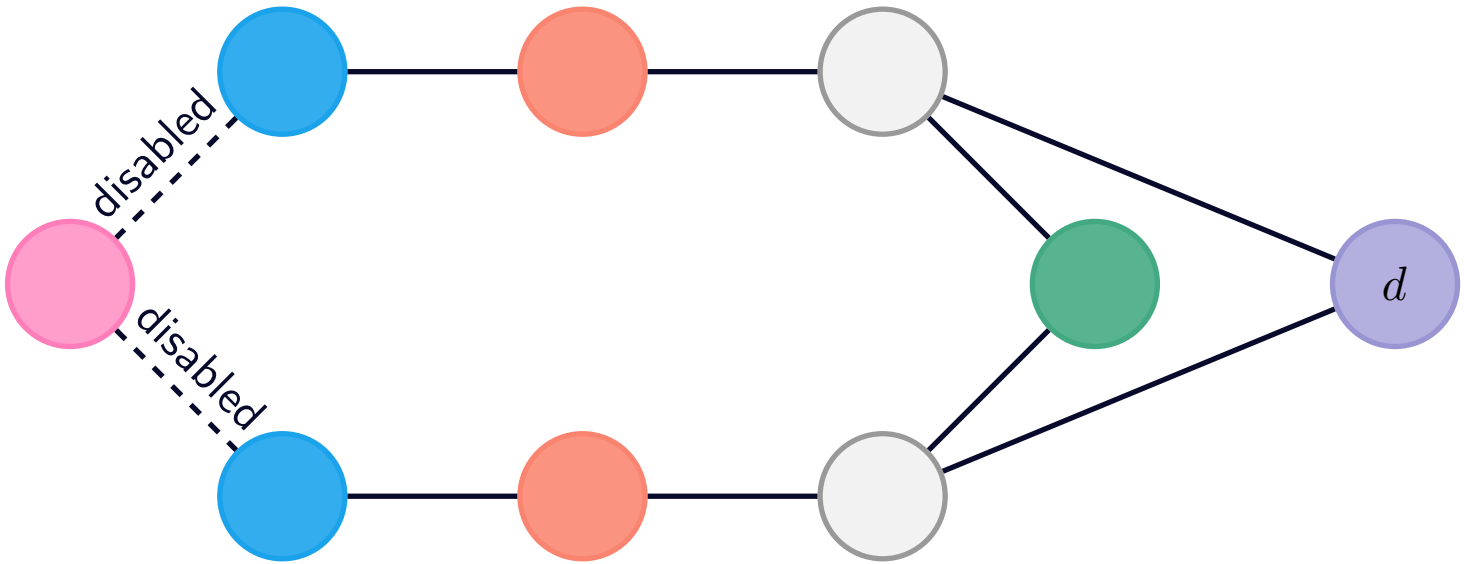
Spurious counterexample $\Rightarrow$ refine the abstraction.

Counterexample-guided refinement



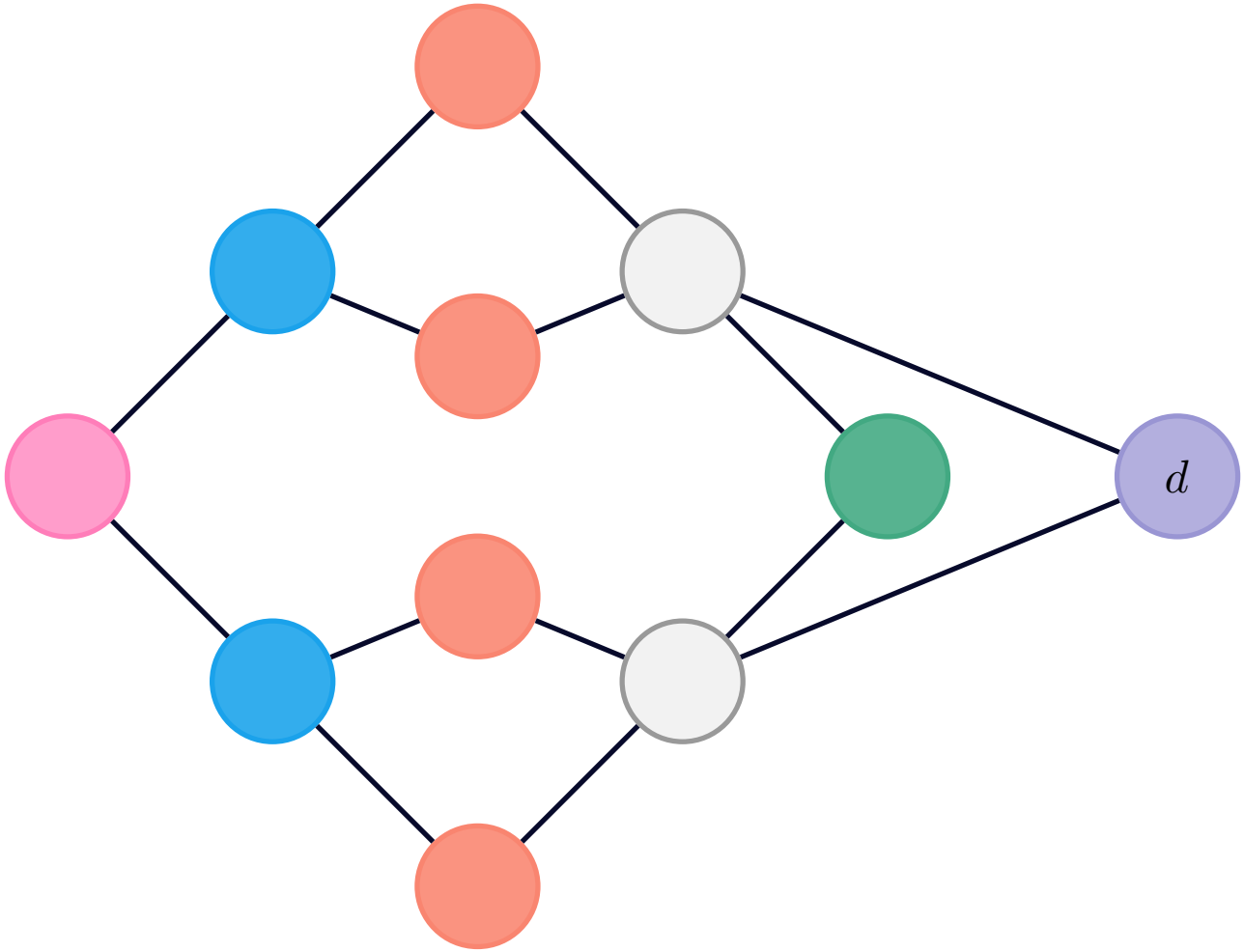
No progress!

Counterexample-guided refinement



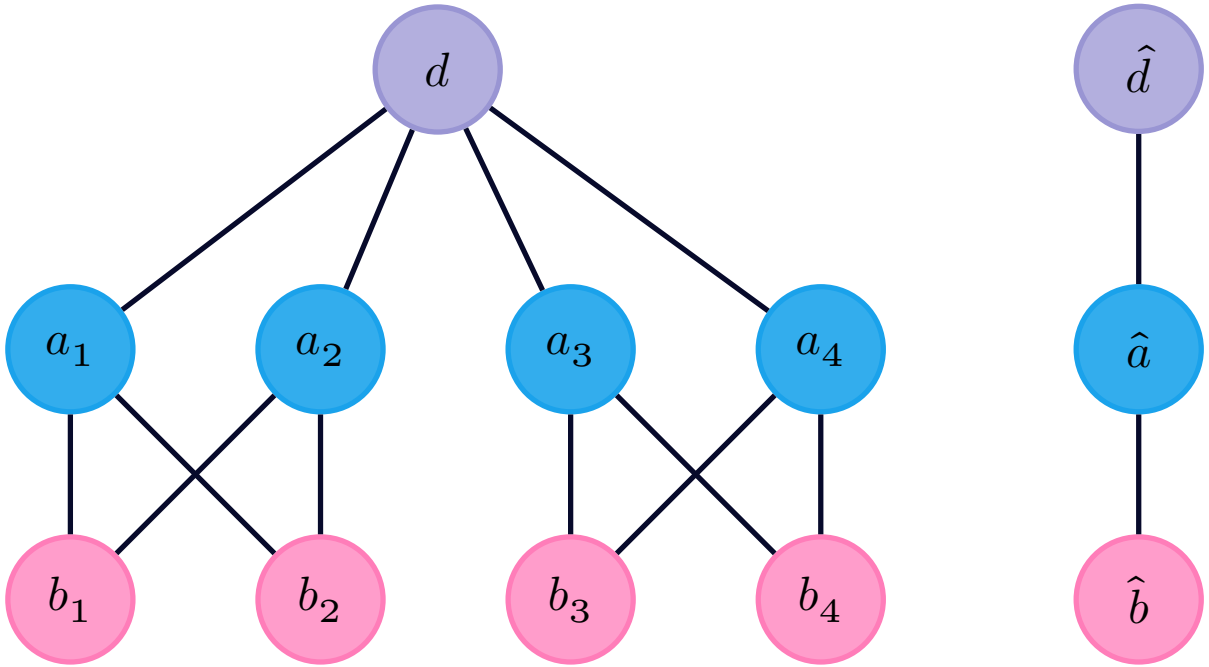
- *Learned* that pink nodes do not send routes to blue nodes.
- Start over, `REFINE` until $|\mathrm{mincut}(\text{Graph-disabled,blue})| > 1$.

Counterexample-guided refinement



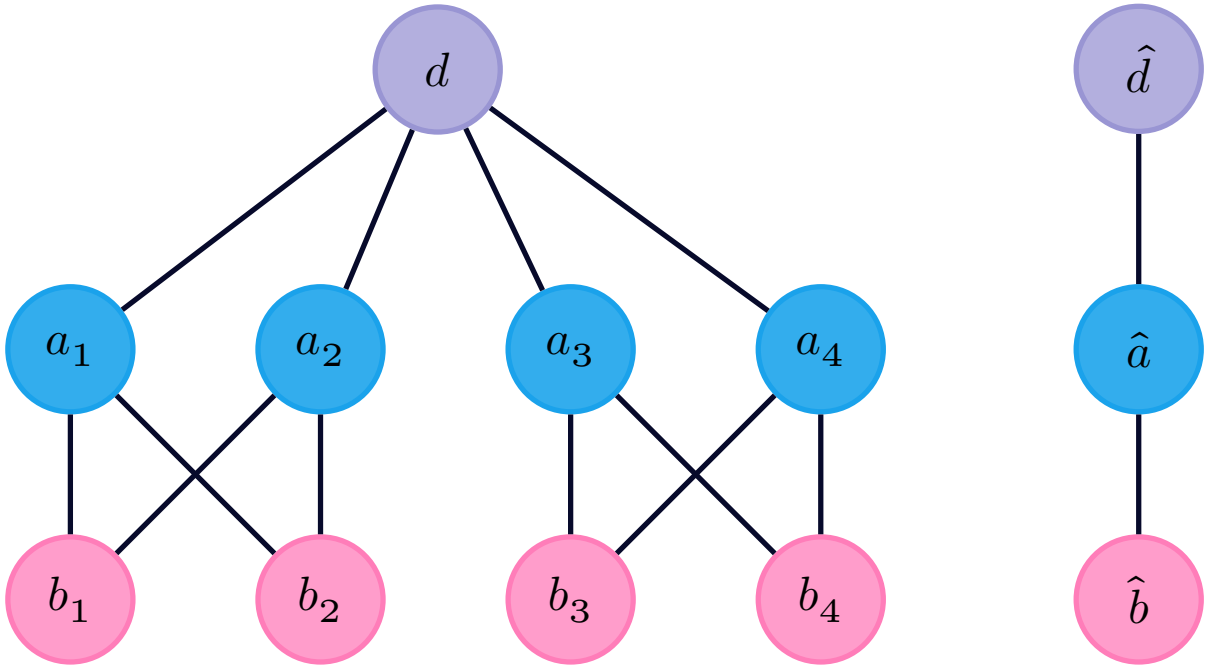
- Verifies reachability under any single link failure.
- Carries over to the concrete network by soundness theorem!

The REFINE procedure



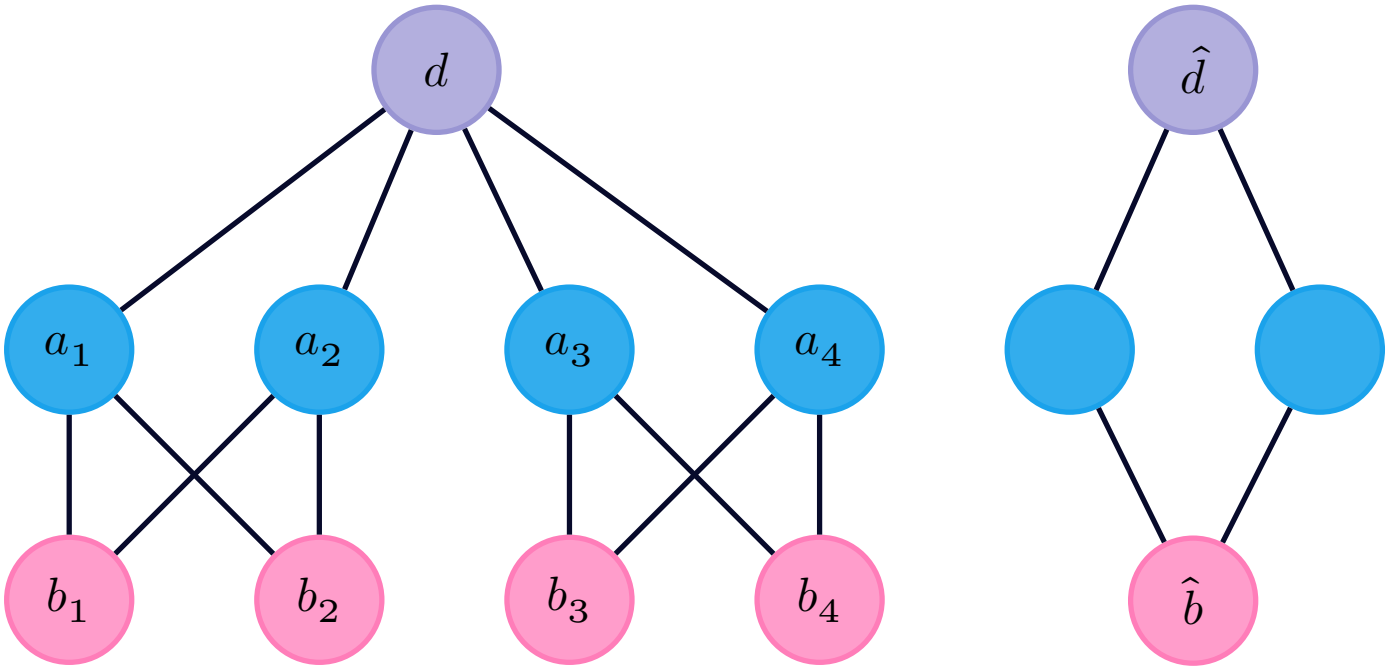
- **Goal:** Compute a plausible abstraction.

The REFINE procedure



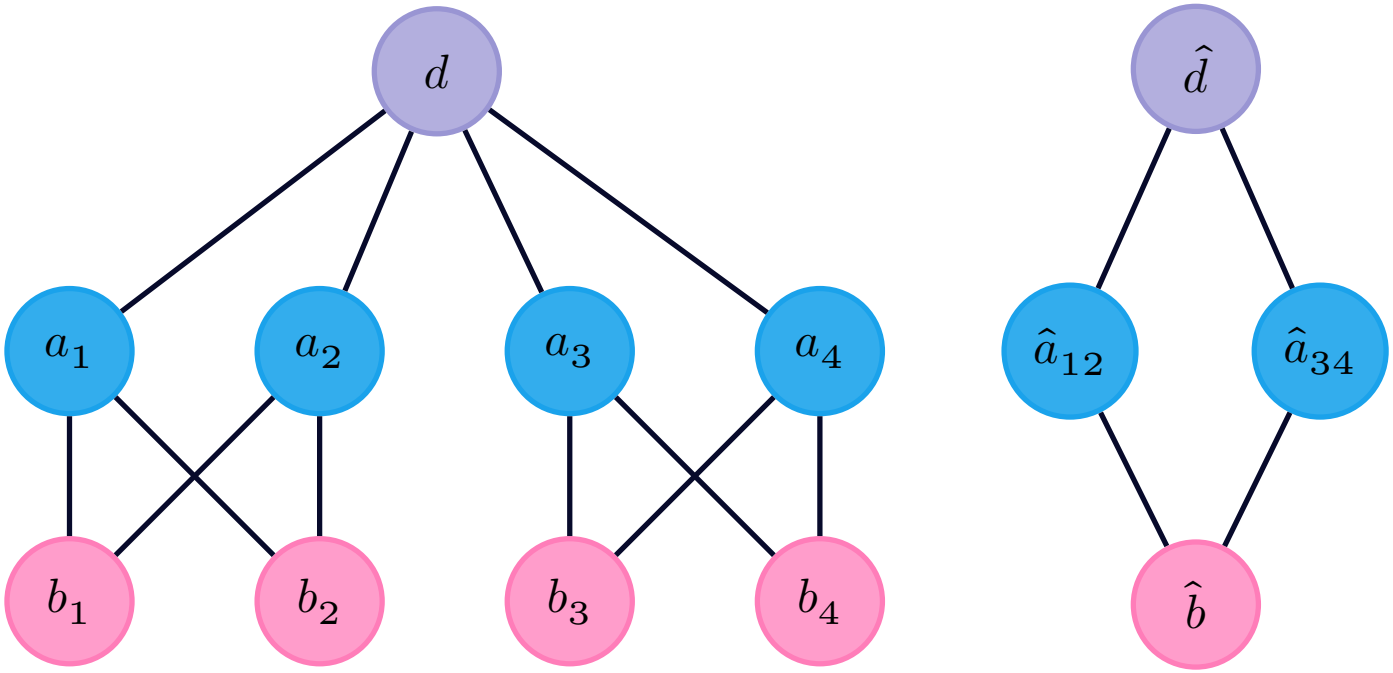
- **Goal:** Compute a plausible abstraction.
- Split abstract nodes, but:
  (1) *Which* nodes to split?

The REFINE procedure



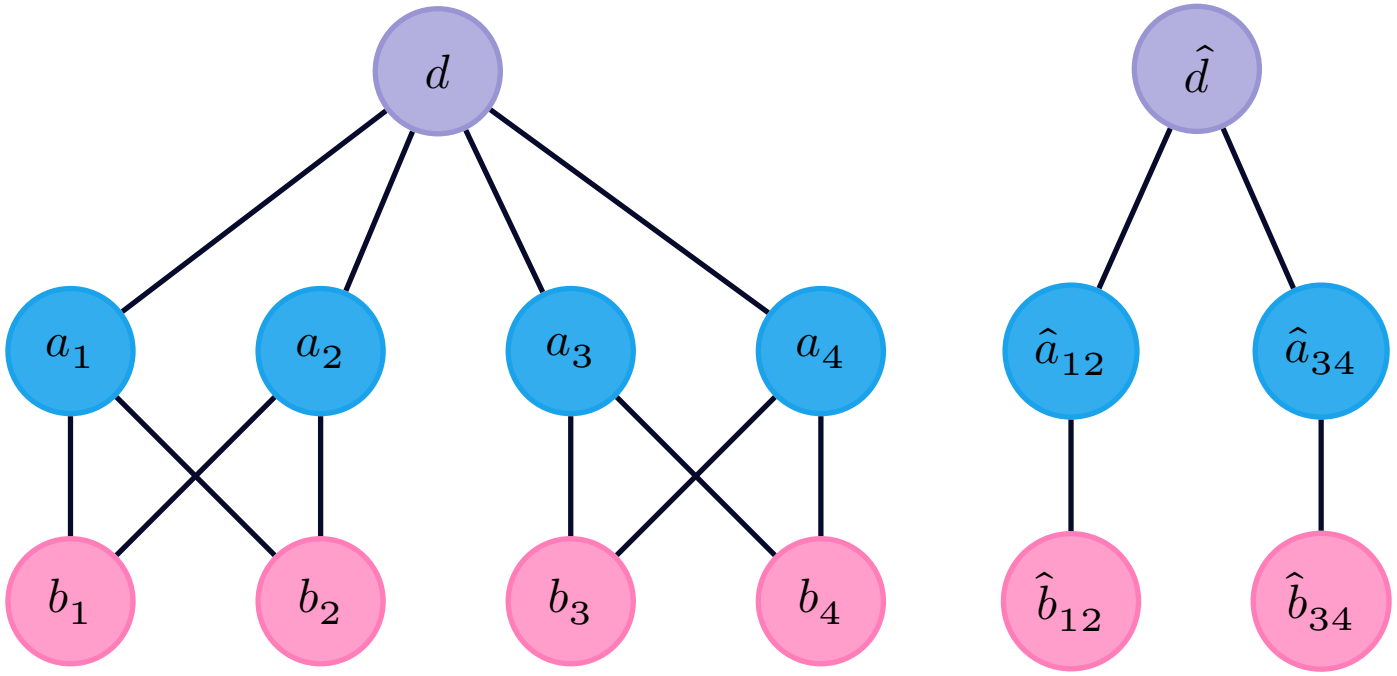
- **Goal:** Compute a plausible abstraction.
- Split abstract nodes, but:
  (1) *Which* nodes to split?

The REFINE procedure



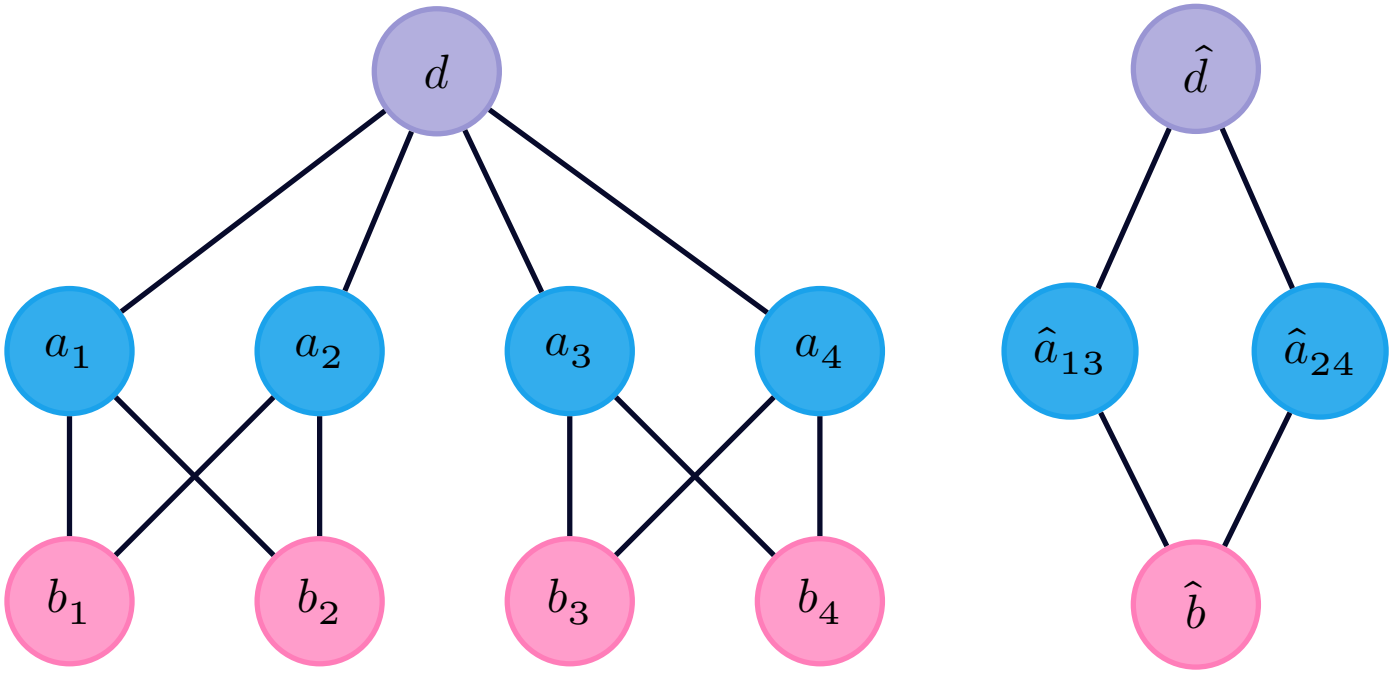
- **Goal:** Compute a plausible abstraction.
- Split abstract nodes, but:
  (1) *Which* nodes to split?
  (2) *How* to split them?

The REFINE procedure



- **Goal:** Compute a plausible abstraction.
- Split abstract nodes, but:
  (1) *Which* nodes to split?
  (2) *How* to split them?
  (3) Must remain a valid $\forall\exists$-abstraction.

The REFINE procedure



- **Goal:** Compute a plausible abstraction.
- Split abstract nodes, but:
  (1) *Which* nodes to split?
  (2) *How* to split them?
  (3) Must remain a valid $\forall\exists$-abstraction.
  (4) Need to make the right splitting choices.

The `REFINE` procedure

- Computing the smallest plausible abstraction seems difficult!
- Instead: Explore many plausible abstractions.
- Guide the search by a set of heuristics.
- Pick the smallest abstraction found.

Evaluation

Compression and Verification results

Topo	V/E	# Failed	Abs V/E	Abstraction Time	SMT Time
FT20	500/8000	1	9/20	0.1	0.1
		3	40/192	1.0	7.6
		5	96/720	2.5	248
FT40	2000/64000	1	12/28	0.1	0.1
		3	45/220	33	12.3
		5	109/880	762.3	184.1

- Evaulated on synthetic datacenter topologies.
- Often reduced edges by more than 100x.
- Abstraction time is insignificant.
- SMT verification is possible.

Compression and Verification results

Topo	V/E	# Failed	Abs V/E	Abstraction Time	SMT Time
FT20	500/8000	1	9/20	0.1	0.1
		3	40/192	1.0	7.6
		5	96/720	2.5	248
FT40	2000/64000	1	12/28	0.1	0.1
		3	45/220	33	12.3
		5	109/880	762.3	184.1

- Evaulated on synthetic datacenter topologies.
- Often reduced edges by more than 100x.
- Abstraction time is insignificant.
- SMT verification is possible.

Compression and Verification results

Topo	V/E	# Failed	Abs V/E	Abstraction Time	SMT Time
FT20	500/8000	1	9/20	0.1	0.1
		3	40/192	1.0	7.6
		5	96/720	2.5	248
FT40	2000/64000	1	12/28	0.1	0.1
		3	45/220	33	12.3
		5	109/880	762.3	184.1

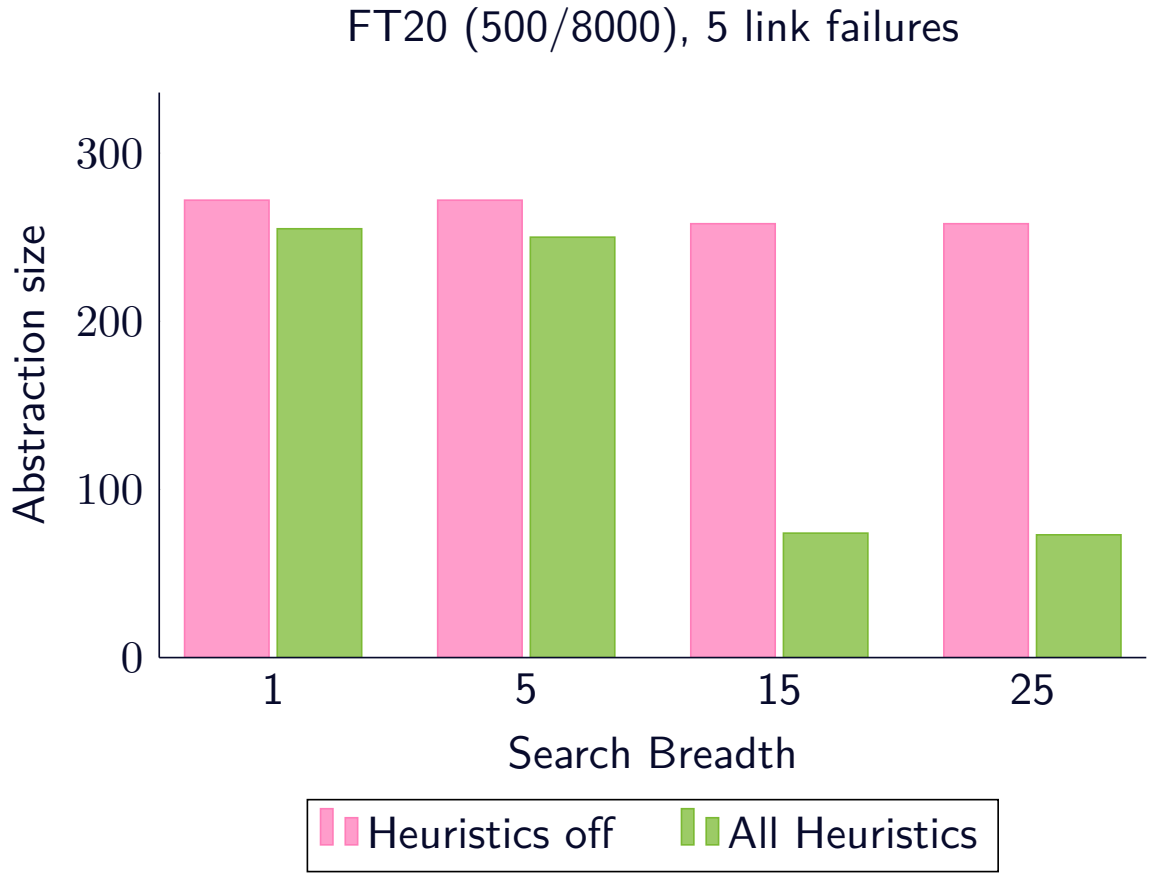
- Evaulated on synthetic datacenter topologies.
- Often reduced edges by more than 100x.
- Abstraction time is insignificant.
- SMT verification is possible.

Compression and Verification results

Topo	V/E	# Failed	Abs V/E	Abstraction Time	SMT Time
FT20	500/8000	1	9/20	0.1	**0.1**
		3	40/192	1.0	**7.6**
		5	96/720	2.5	**248**
FT40	2000/64000	1	12/28	0.1	**0.1**
		3	45/220	33	**12.3**
		5	109/880	762.3	**184.1**

- Evaulated on synthetic datacenter topologies.
- Often reduced edges by more than 100x.
- Abstraction time is insignificant.
- SMT verification is possible.

Heuristics effectiveness



- Random searches will not achieve high compression.
- Heuristics make (costly) mistakes.

Conclusions

We enable verification of fault tolerance of large networks:

- Based on a new theory of network compression.
- Origami a tool that can handle networks out of reach to current state-of-the-art tools.
- Geared towards reachability only.
- Some properties are not preserved by approximation.

Thank you!