IEEE INFOCOM

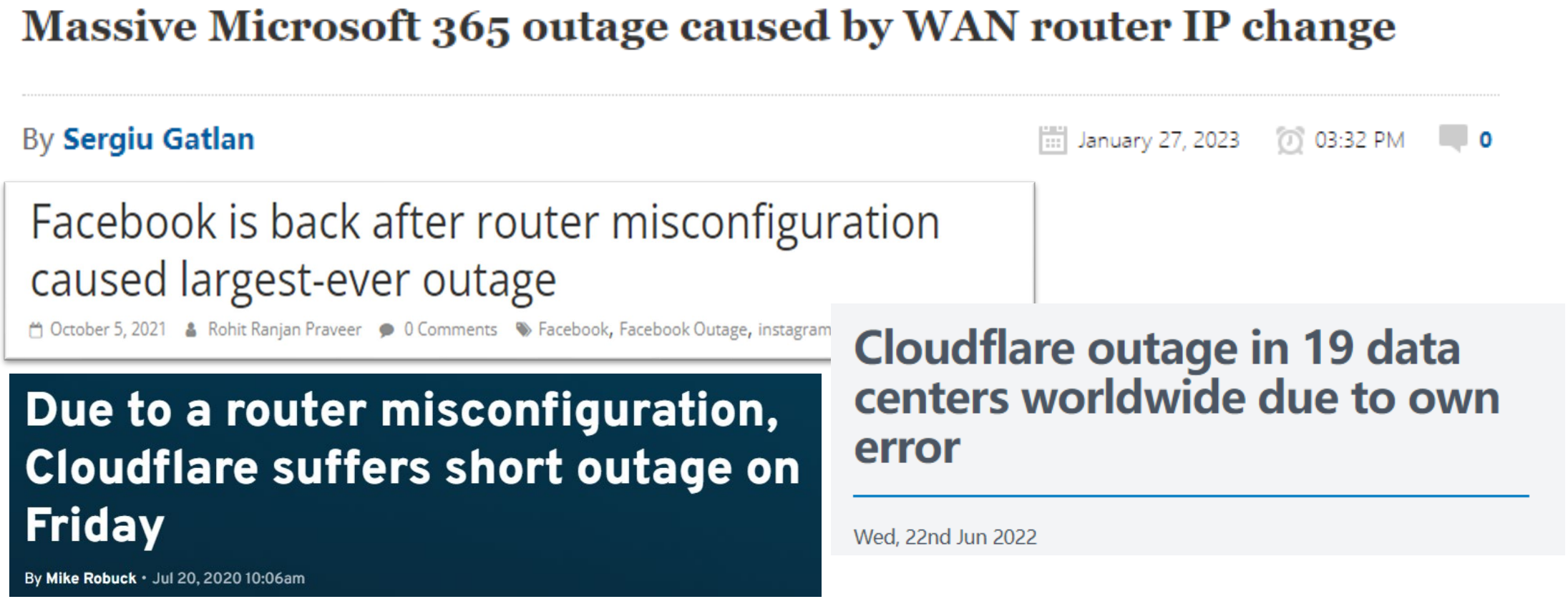
Network Can Help Check Itself: Accelerating SMT-based Network Configuration Verification Using Network Domain Knowledge

Xing Fang[1], Feiyan Ding[1], Bang Huang[1], Ziyi Wang[1], Gao Han[1], Rulan Yang[1], Lizhao You[1], Qiao Xiang[1], Linghe Kong[2], Yutong Liu[2], Jiwu Shu[1, 3]

[1] Xiamen University, [2] Shanghai Jiao Tong University, [3] Minjiang University

2024/5/23

Network Configuration Errors Are Common



Massive Microsoft 365 outage caused by WAN router IP change

By Sergiu Gatlan

January 27, 2023 03:32 PM 0

Facebook is back after router misconfiguration caused largest-ever outage

October 5, 2021 Rohit Ranjan Praveer 0 Comments Facebook, Facebook Outage, instagram

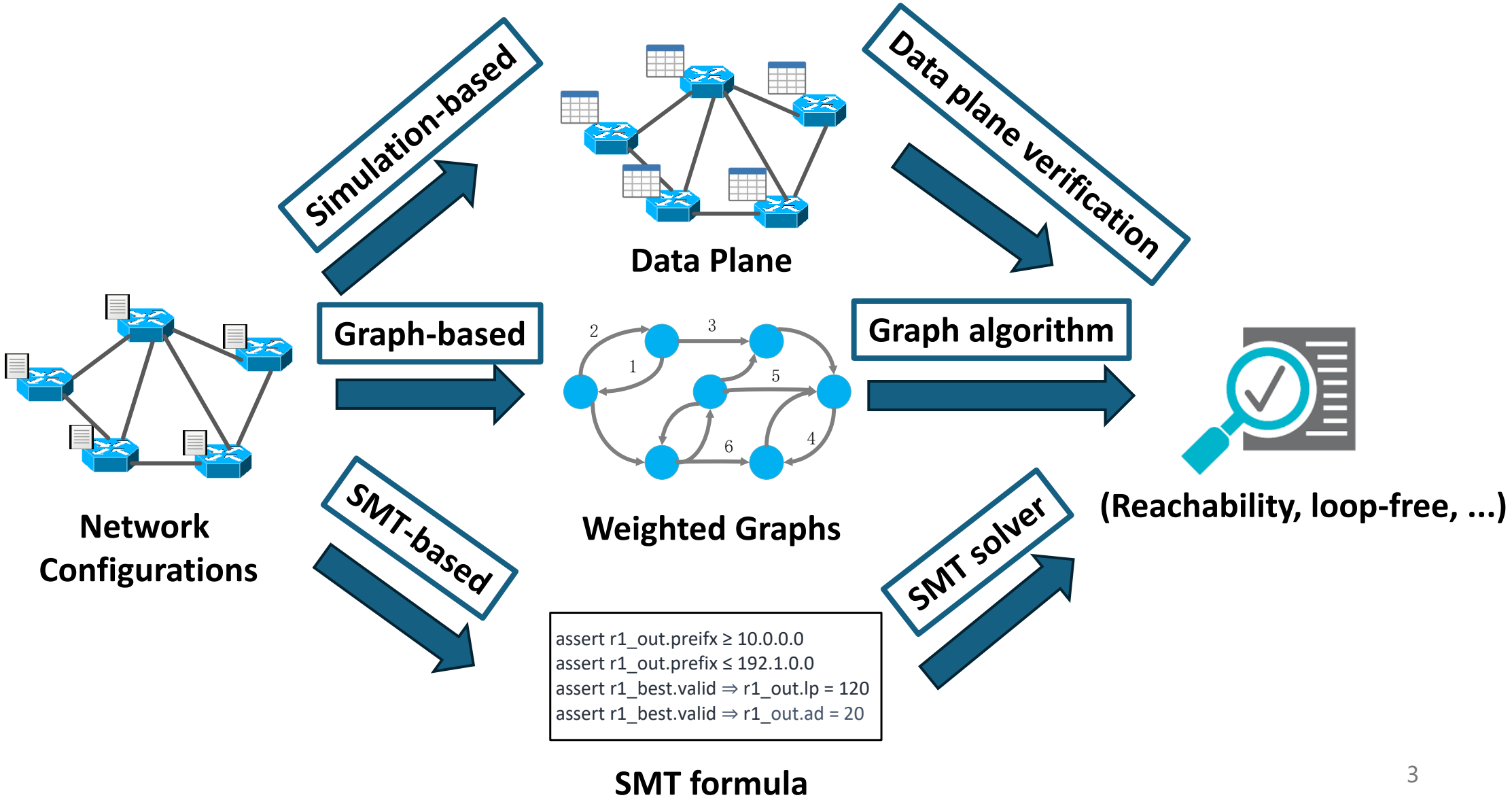
Due to a router misconfiguration, Cloudflare suffers short outage on Friday

By Mike Robuck · Jul 20, 2020 10:06am

Cloudflare outage in 19 data centers worldwide due to own error

Wed, 22nd Jun 2022

Configuration Verification Can Ensure Correctness



Limitation of Existing Techniques

Simulation-based

Batfish (NSDI ’15)
Hoyan (SIGCOMM ’20)
SRE (SIGCOMM ’22)
DNA (NSDI ’22)

cannot handle multiple convergences

Graph-based

ARC (SIGCOMM ’16)
Tiramisu (NSDI ’20)

cannot model complex protocol features

SMT-based

Bagpipe (OOPSLA ’16)
Minesweeper (SIGCOMM ’17)
BiNode (INFOCOM ’20)

cannot scale

Limitation of Existing Techniques

Simulation-based

Batfish (NSDI ’15)
Hoyan (SIGCOMM ’20)
SRE (SIGCOMM ’22)
DNA (NSDI ’22)

cannot handle multiple convergences

Graph-based

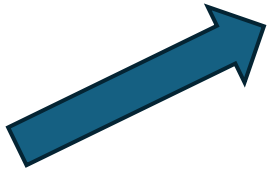
ARC (SIGCOMM ’16)
Tiramisu (NSDI ’20)

cannot model complex protocol features

SMT-based

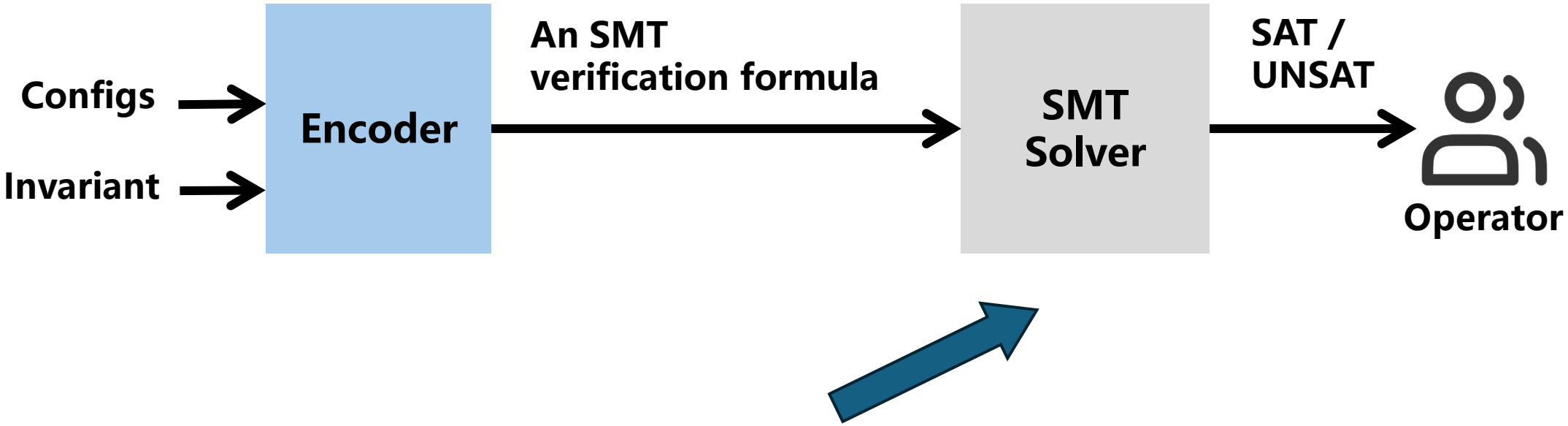
Bagpipe (OOPSLA ’16)
Minesweeper (SIGCOMM ’17)
BiNode (INFOCOM ’20)

cannot scale



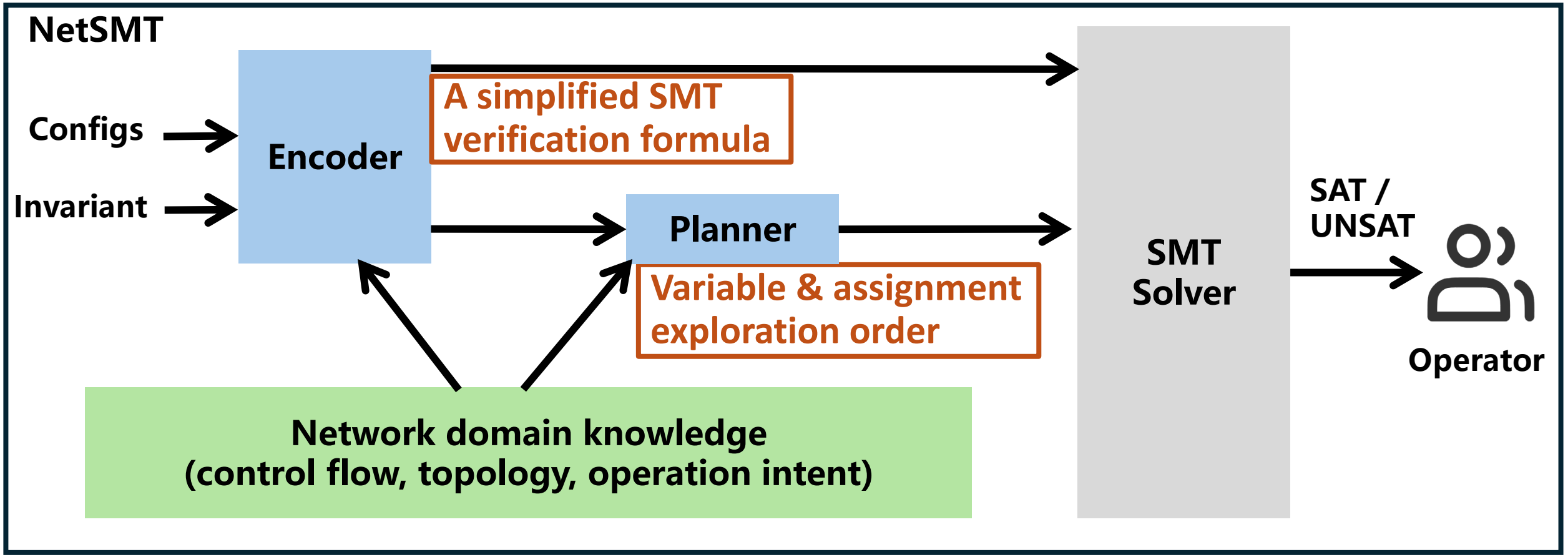
This paper aims to solve

Motivation



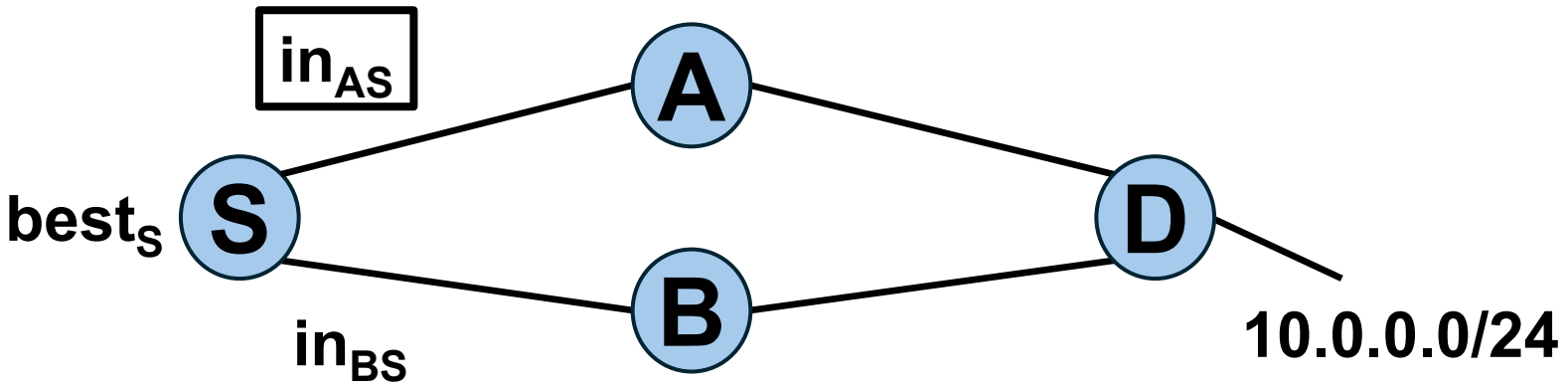
Lacks network domain knowledge

Our Approach: NetSMT



Using network domain knowledge to guide the SMT solving

SMT-based Verification



Network model example (route selection)

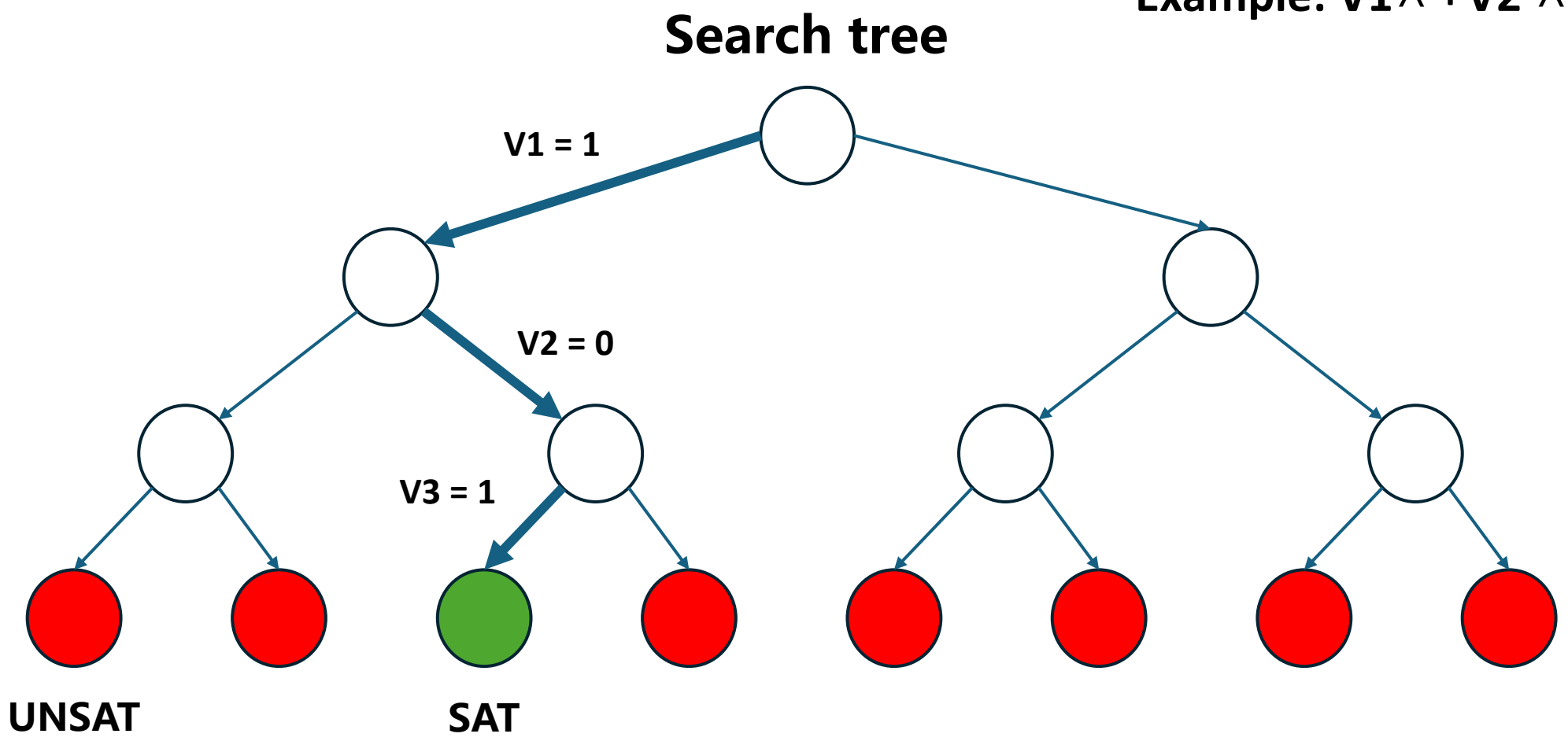
$(best_S = in_{AS} \vee best_S = in_{BS}) \wedge best_S \leq in_{AS} \wedge best_S \leq in_{BS}$

SMT formula: $N \wedge \neg P$

Goal: search a **satisfiable variable assignment** (a stable data plane that violate the invariant)

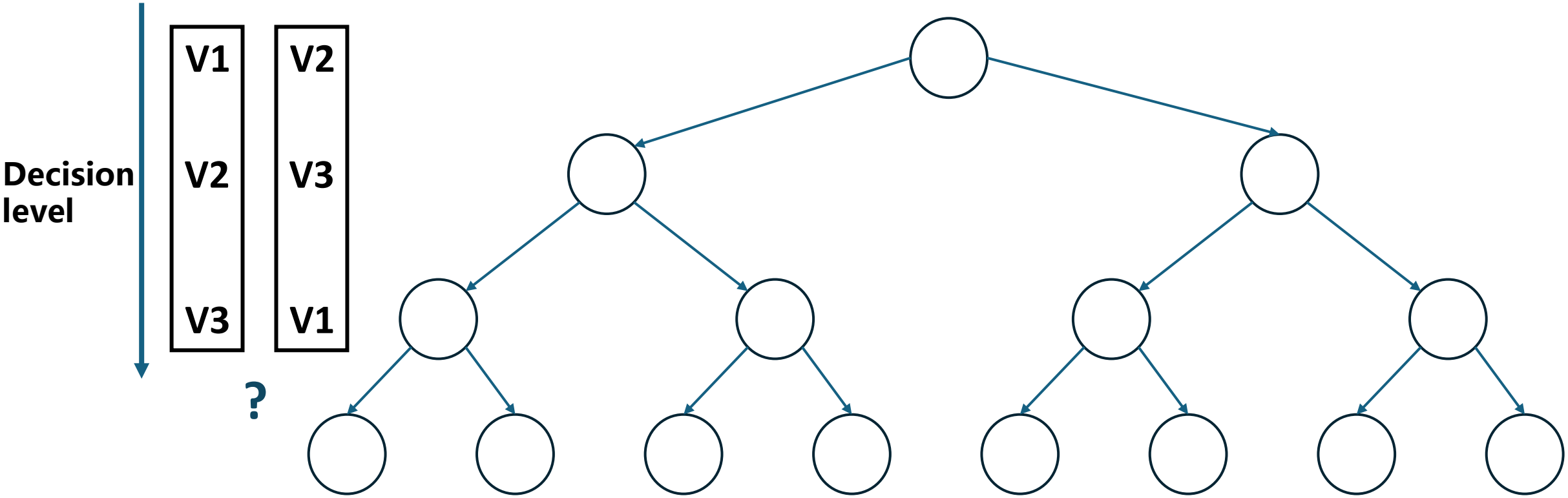
Key Insight

Example: V1∧¬ V2 ∧V3



Guide the search process with network domain knowledge

Guide Variable Exploration Order



Which Boolean variable should be explored first?

Guide Variable Exploration Order

if out_{AS}**.prefix** == 10.0.0.0/24

then

in_{AS}.valid = true — V1

in_{AS}.comm910 = true — V2

in_{AS}.lp = 100 — V3

in_{AS}.med = 100 — V4

…

2^4 = 16 assignments

1 → SAT

15 → **UNSAT**

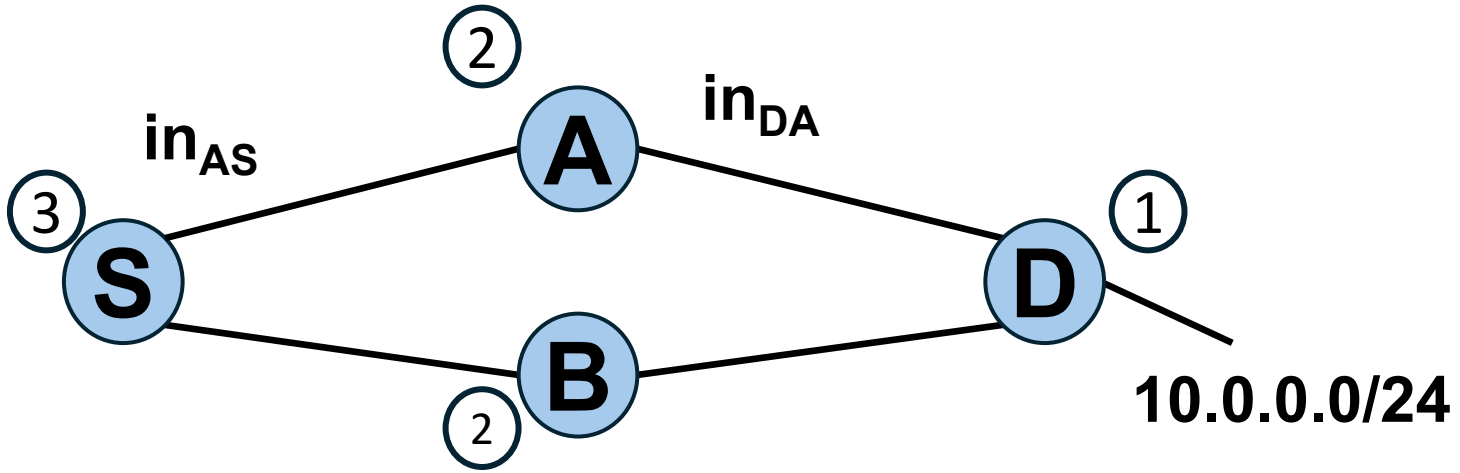
(Pruned by first explore branching variables)

[1] **Guideline 1 :** All branching variables are prior to other variables.

[1] Chen J, He F. Control flow-guided SMT solving for program verification. ASE 2018.

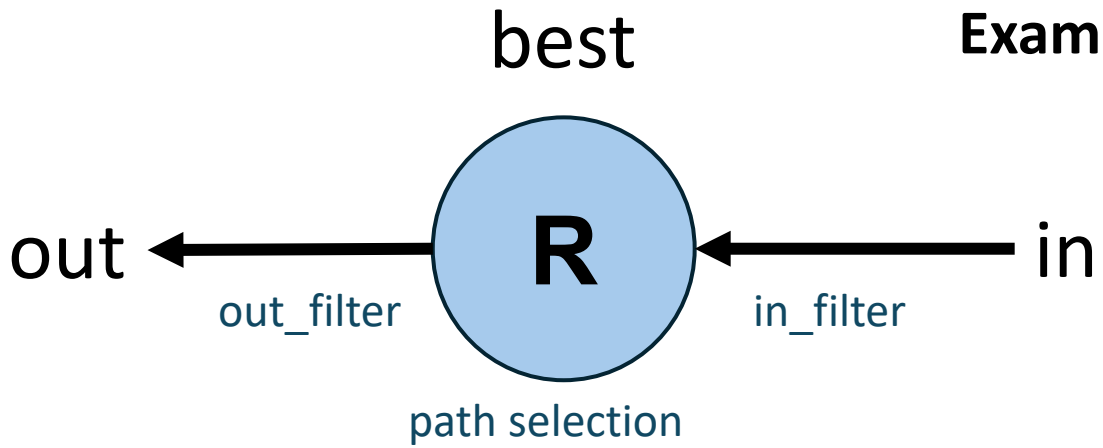
Guide Variable Exploration Order

Intuition: simulate the flooding behavior of BGP
Example: explore in_{DA} before in_{AS}



Guideline 2: Branching variables whose residing router is closer to the destination are explored first

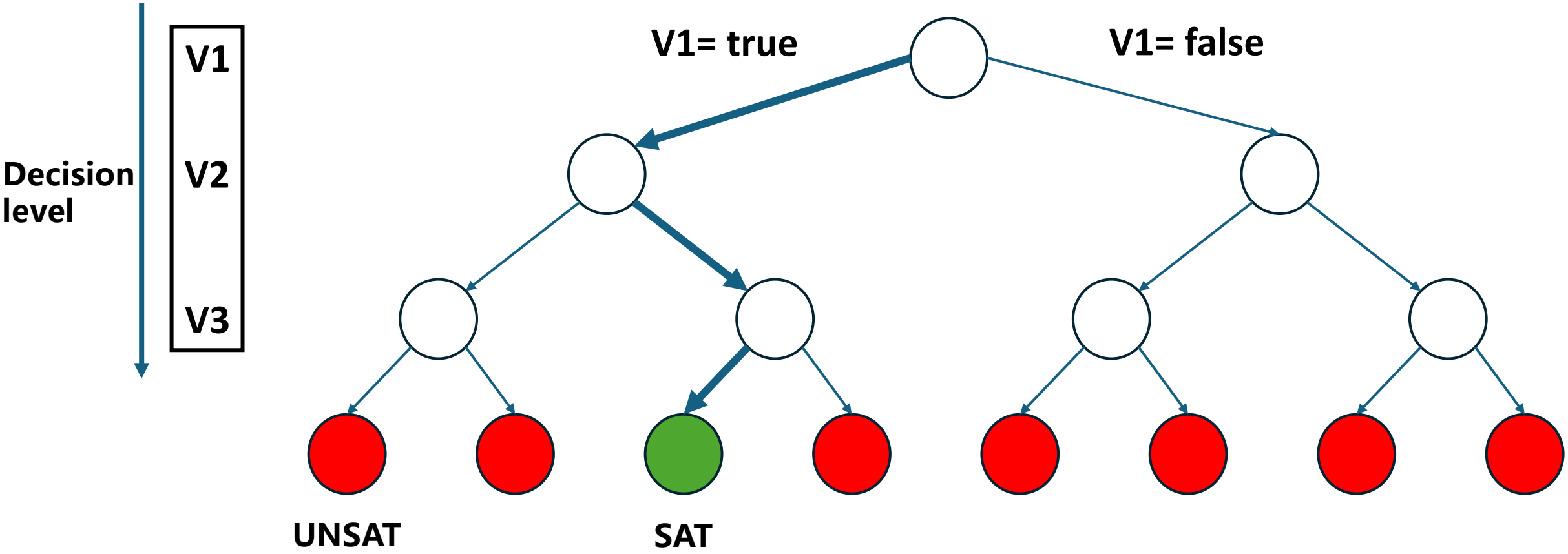
Guide Variable Exploration Order



Intuition: simulate BGP Route Processing Procedure
Example: in < best < out

Guideline 3: For branching variables residing in the same router, order them based on route announcement type

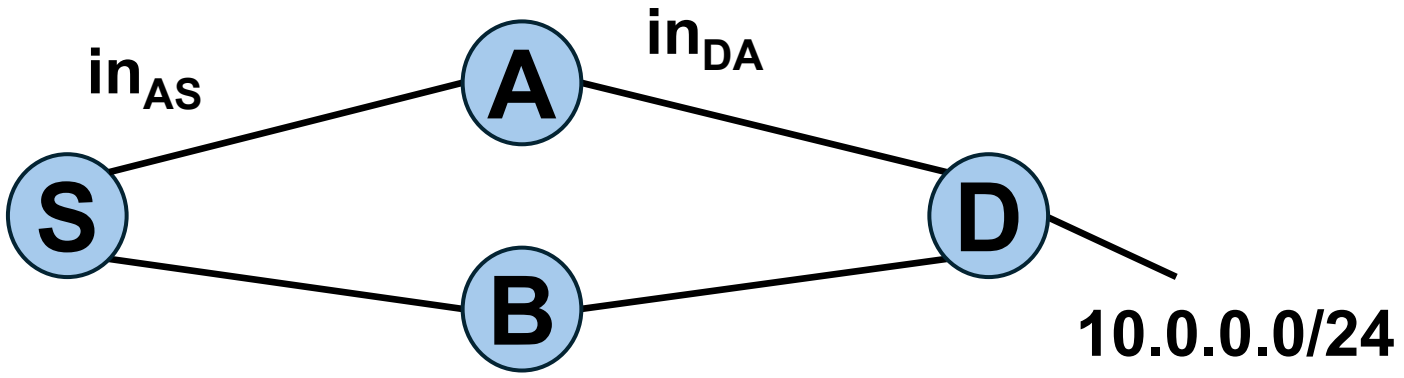
Guide Assignment Exploration Order



Boolean variables should be assigned true or false?

Guide Assignment Exploration Order

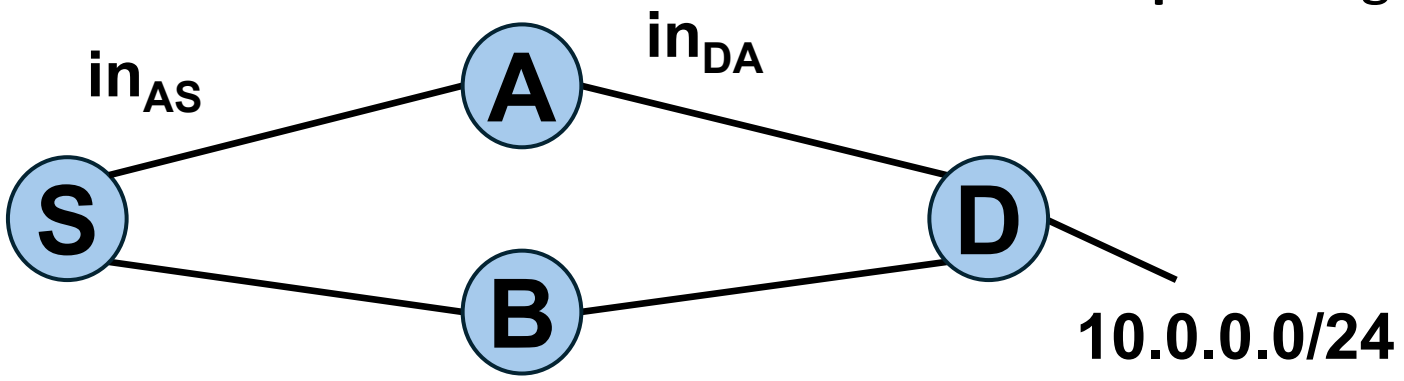
Intent: S can reach **10.0.0.0/24**
Example: assign in_{AS} and in_{DA} to **true**



Guideline 4: For branching variables, we prefer the value assignment that is consistent with operating intent.

Guide Assignment Exploration Order

Intent: S can reach **10.0.0.0/24**
suspicious error: router S deny routes
Example: assign in_{AS} to **false**



Guideline 5: For branching variables that reside in the suspicious error routers, we prefer the assignment that is contrary to intent

Simplify SMT formula encoding

if out_{DA}.prefix == 10.0.0.0/24
then
in_{DA}.valid = true
in_{DA}.comm910 = true
in_{DA}.comm920 = out_{DA}.comm920
…
else if out_{DA}.prefix == 10.1.0.0/24
then
in_{DA}.valid = true
in_{DA}.comm910 = out_{DA}.comm910
in_{DA}.comm920 = true
…

Invariant: S can reach **10.0.0.0/24**

Pruned due to irrelevance to the invariant

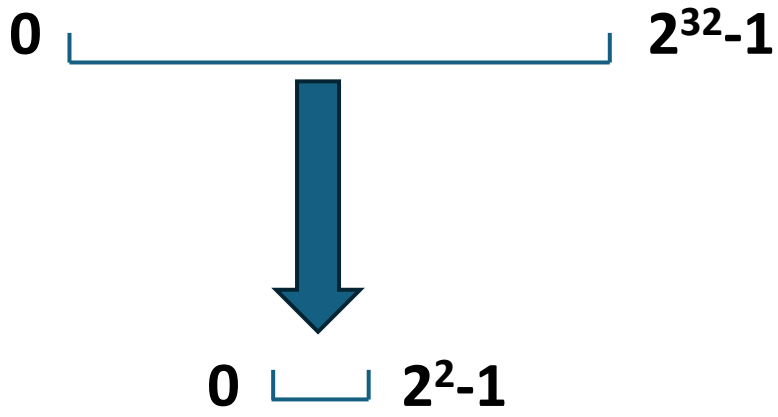
Prune unrelated configurations based on the invariant

Simplify SMT formula encoding

router S

policy **A to S:**
if community == 910
  set local-preference 50
if community == 920
  deny
policy **B to S:**
if community == 930
  set local-preference 200
if community == 940
  permit

local-preference can only be assigned to 50，200，100 (default)

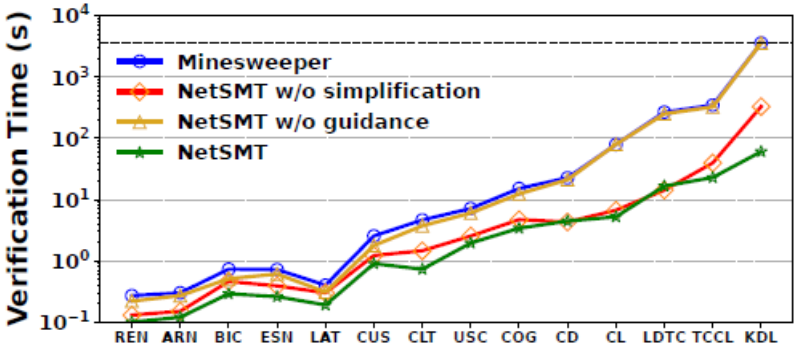


Replace the concrete value with the abstract value

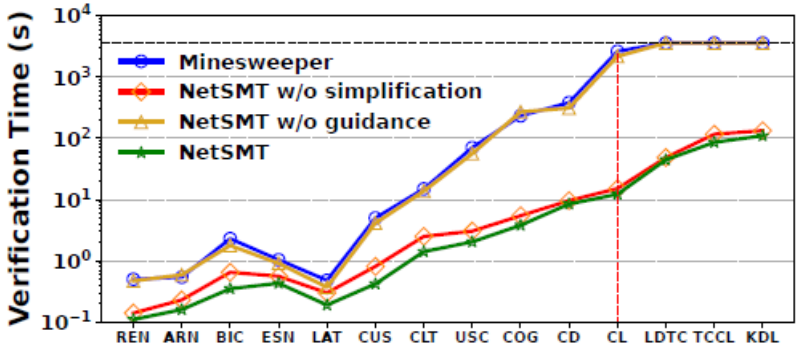
Evaluation

- Implementation
  - Implement guided SMT solving based on Z3-4.12.2
  - Implement SMT formula simplification based on Minesweeper
- Experiment Setting
  - WAN (34 to 755) and DCN (Fat-tree, k = 4~20) topologies
  - Synthesized configurations
  - Properties: Pair-wise reachability/isolation with/without k failure, forwarding table computation
  - Compared with Minesweeper (SIGCOMM ’17) and BiNode (INFOCOM ’20)

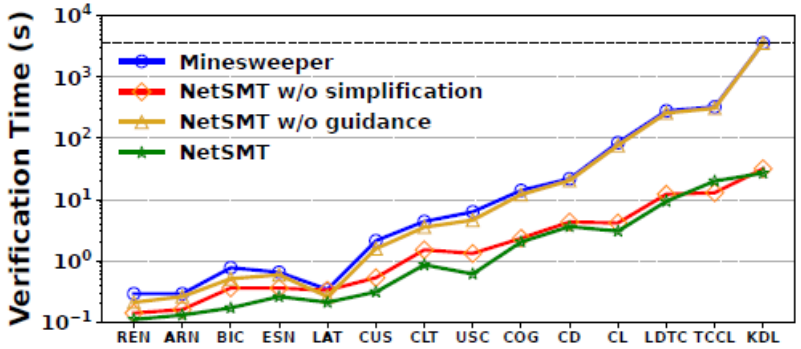
Evaluation



(a) reachability and isolation



(b) reachability and isolation with k-link-failure



(c) forwarding

Fig. 7. The SMT verification time on the satisfiable benchmarks on WAN.

- **WAN topology: 34-755 nodes**
- **NetSMT offers up to 215.8x acceleration compared to Minesweeper**

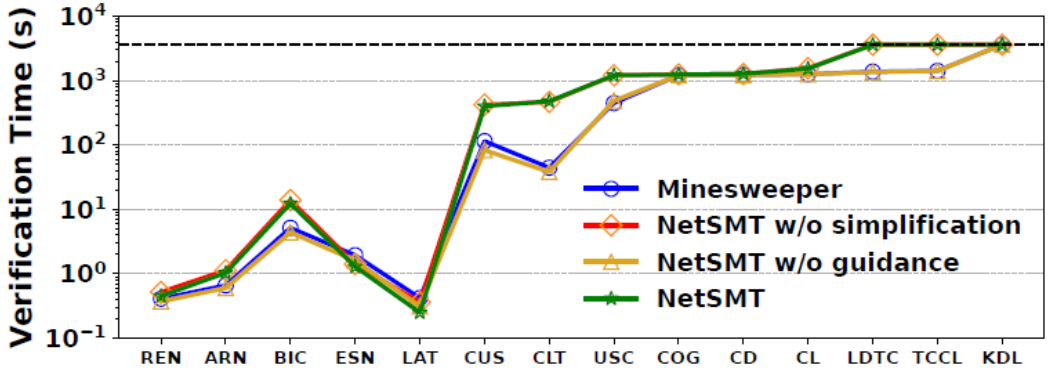
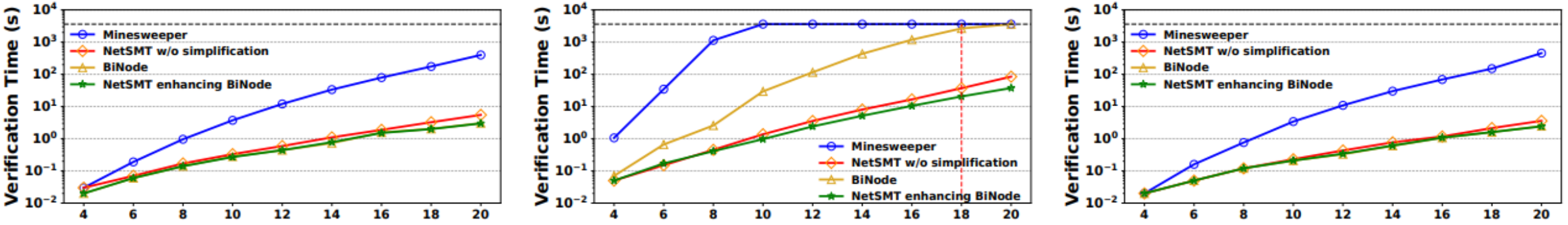


Fig. 9. The SMT verification time on the unsatisfiable benchmarks on WAN.

Evaluation



(a) reachability and isolation (b) reachability and isolation with k-link-failure (c) forwarding

Fig. 10. The SMT verification time on the satisfiable benchmarks on DCN.

- DCN topology: Fat-tree (k = 4~20)
- NetSMT offers up to 129.5x acceleration compared to BiNode

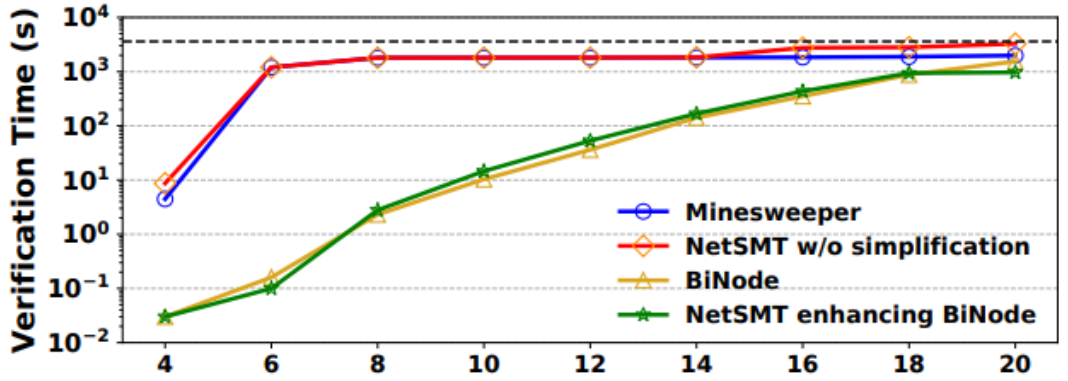
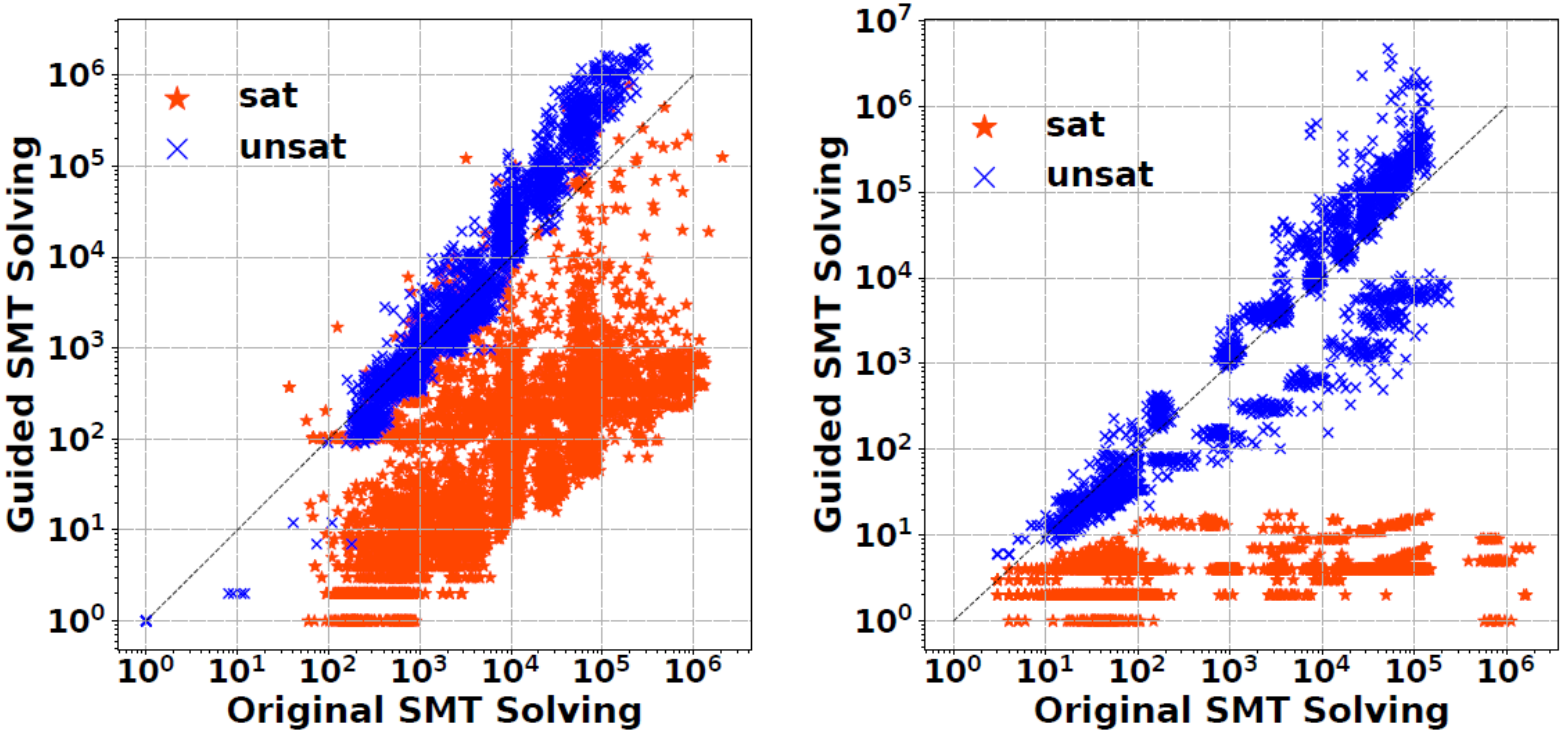


Fig. 12. The SMT verification time on the unsatisfiable benchmarks on DCN.

Evaluation



(a) number of conflicts on WAN (b) number of conflicts on DCN

Fig. 8. Number of conflicts on all benchmarks.

NetSMT can find the satisfiable assignment with less conflicts

Conclusion

- NetSMT: a SMT-based CPV
  - Guide variable & assignment exploration order
  - Simplify SMT verification formula
- Extensive evaluation
- Open-source
  - **https://github.com/sngroup-xmu/NetSMT**