

The voice of the flow: A graph-based approach for step-wise explanations of CSPs

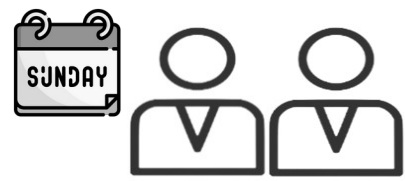
NGUYEN Minh Anh

MAI Anh Tien

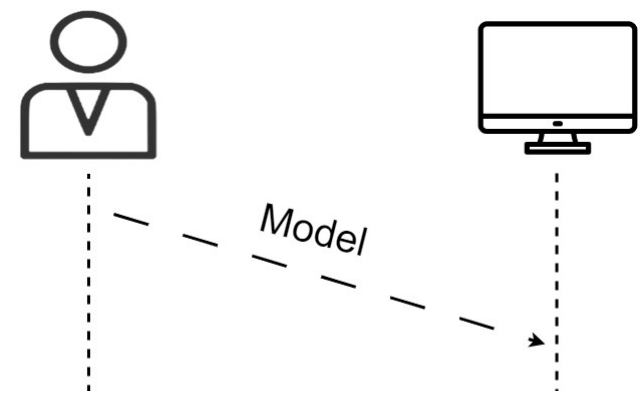
LAU Hoong Chuin

Why Did It Fail? Nobody Knows

Nurses: I'm sick

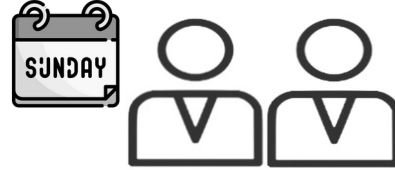


You (scheduler):

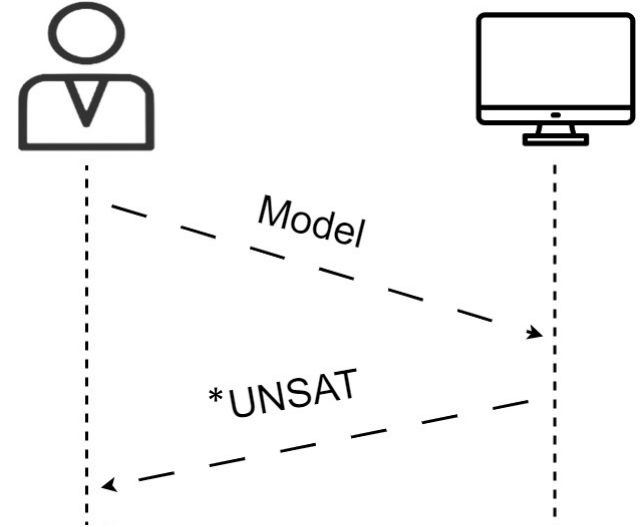


Why Did It Fail? Nobody Knows

Nurses: I'm sick



You (scheduler):

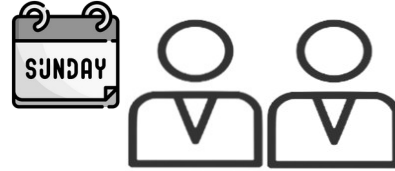


INFEASIBLE - NO SOLUTION

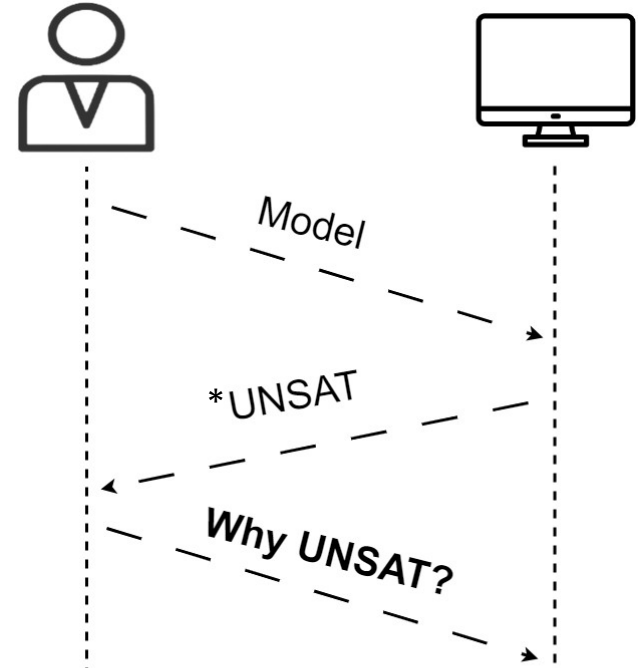
*UNSAT = unsatisfiable

Why Did It Fail? Nobody Knows

Nurses: I'm sick



You (scheduler):



INFEASIBLE - NO SOLUTION

WHY ???

*UNSAT = unsatisfiable

Explaining Overconstrained Problems

solver:

UNSAT

user:

Why UNSAT?

explainer:

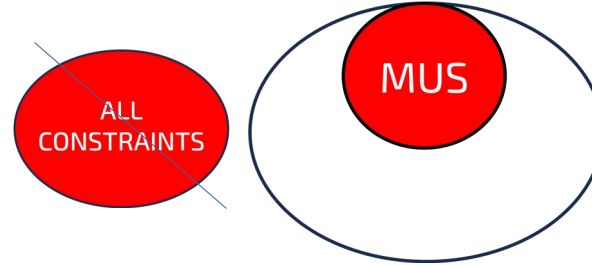


Explaining Overconstrained Problems

solver:
user:

UNSAT
Why UNSAT?

explainer:



(Source [1])

Minimal Unsatisfiable Subsets (MUS)

minimal set constraints causing a conflict
... ease cognitive burden for user

Easy MUSs

Length of MUS = 3

- (C1) Robert has a day off on Tue
- (C2) Shift D on Tue must be covered 7/8 nurses
- (C3) Kristen has a day off on Tue

Reasoning:

Step 1: (C1) and (C2) $\rightarrow$ (Kristen,Tue)=D (C4)

Step 2: (C3) and (C4) $\rightarrow$ **Conflict !**

Nurse	Tue		Tue		Tue
Megan			D		D
Robert	-		-		-
Kevin			D		D
Amy			D		D
Katherine		Step 1 $\rightarrow$	D	Step 2 $\rightarrow$	D
Kristen			D		D
Jonathan			D		D
Richard			D		D
Cover D	7/8		7/8		7/8

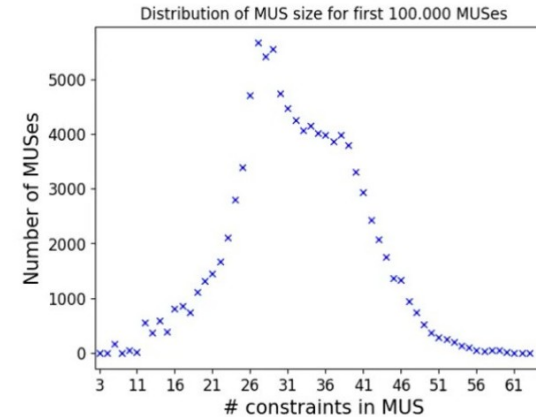
conflict with (C3)
(Kristen,Tue) $\neq$ D

MUSs are not trivial to comprehend sometimes

Not-so-easy MUSes:

Length of MUS: 11

- Shift D on Sat 2 must be covered by 6 nurses out of 8
 - Kevin should have at least 2 consecutive days off
 - Katherine should work at most 1 weekends
 - Shift D on Sun 1 must be covered by 5 nurses out of 8
 - Robert has a day off on Tue 2
 - Kristen should work at most 1 weekends
 - Megan should work at most 1 weekends
 - Richard should work at most 1 weekends
 - Shift D on Tue 2 must be covered by 7 nurses out of 8
 - Jonathan should work at most 1 weekends
 - Kevin has a day off on Mon 2
-
- Robert requests to work shift D on Wed 1
 - Richard has a day off on Sat 1
 - Shift D on Sat 1 must be covered by 5 nurses out of 8
 - Robert requests to work shift D on Thu 1
 - Robert can work at most 5 days before having a day off
 - Kevin should work at most 1 weekends
 - Robert requests to work shift D on Mon 1
 - Robert requests to work shift D on Tue 1
 - Kevin requests to work shift D on Sat 2
 - Katherine has a day off on Sat 1
 - Robert requests to work shift D on Fri 1



Just **168** constraints, yet
100,000+ MUSes exist (Source [1])

What is a Step-Wise Explanation?

A chain of deductions, each using a **minimal** set of constraints

Nurse	Sat	Sun	Mon	Tue	Wed
Megan					
Robert				o	
Kevin			o		
Amy					
Jonathan					
Cover D	5/5			4/5 (Step 1)	

"o" off

"D" day shift

o/D input literals

o/D derived literals

contradiction

Step 1: (Robert,Tue)=OFF + Need 4/5 nurses on Tue

What is a Step-Wise Explanation?

A chain of deductions, each using a **minimal** set of constraints

Nurse	Sat	Sun	Mon	Tue	Wed
Megan				D	
Robert				O	
Kevin			O	D	
Amy				D	
Jonathan				D	
Cover D		5/5		4/5 (Step 1)	

"O" off
 "D" day shift

O/D input literals
O/D derived literals
 contradiction

Step 1: (Robert,Tue)=OFF + Need 4/5 nurses on Tue



Everybody else works

What is a Step-Wise Explanation?

A chain of deductions, each using a **minimal** set of constraints

Nurse	Sat	Sun	Mon	Tue	Wed
Megan				D	
Robert				o	
Kevin	(Step 2)		o	D	
Amy				D	
Jonathan				D	
Cover D		5/5		4/5	

"o" off

"D" day shift

o/D input literals

o/D derived literals

contradiction

Step 2: (Kevin,Mon)=OFF + (Kevin,Tue)=WORK
 + Off at least 2 days before working

What is a Step-Wise Explanation?

A chain of deductions, each using a **minimal** set of constraints

Nurse	Sat	Sun	Mon	Tue	Wed
Megan				D	
Robert				O	
Kevin	(Step 2)	O	O	D	
Amy				D	
Jonathan				D	
Cover D		5/5		4/5	

"O" off

"D" day shift

O/D input literals

O/D derived literals

contradiction

Step 2: (Kevin,Mon)=OFF + (Kevin,Tue)=WORK
 + Off at least 2 days before working



(Kevin,Sun)=OFF

What is a Step-Wise Explanation?

A chain of deductions, each using a **minimal** set of constraints

Nurse	Sat	Sun	Mon	Tue	Wed
Megan				D	
Robert				o	
Kevin		o	o	D	
Amy				D	
Jonathan				D	
Cover D	(Step 3)	5/5		4/5	

"o" off
 "D" day shift

o/D input literals
o/D derived literals
 contradiction

Step 3: (Kevin,Sun)=OFF + Need 5/5 nurses on Sun



CONFLICT

What is a Step-Wise Explanation?

A chain of deductions, each using a **minimal** set of constraints

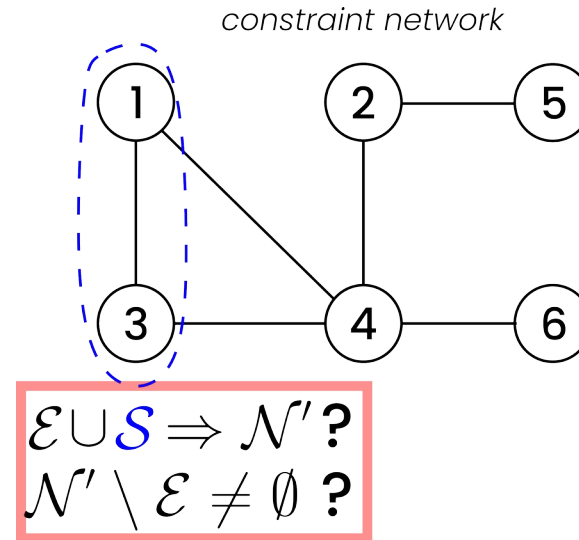
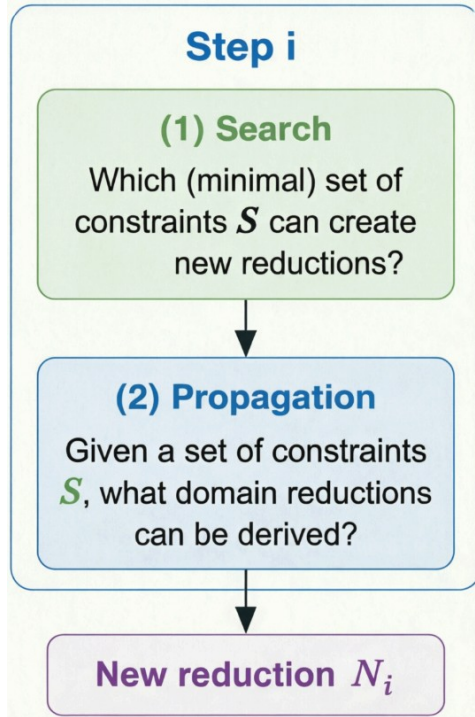
Nurse	Sat	Sun	Mon	Tue	Wed
Megan				D	
Robert				O	
Kevin	(Step 2)	O	O	D	
Amy				D	
Jonathan				D	
Cover D	(Step 3)	5/5		4/5	(Step 1)

"O" off
 "D" day shift

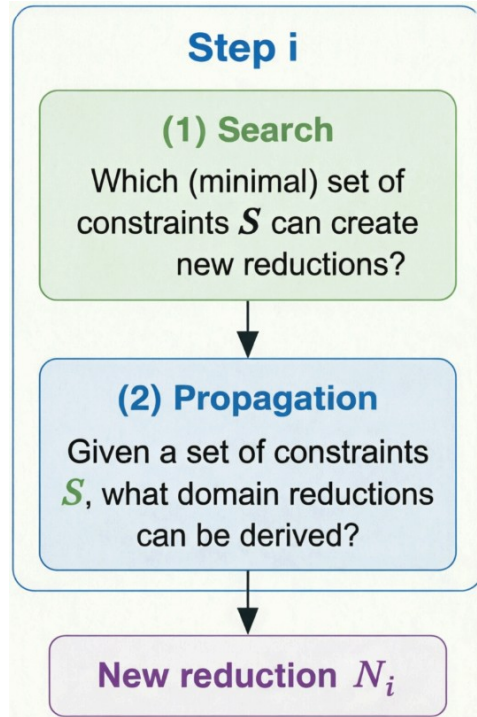
O/D input literals
O/D derived literals
 contradiction

$$\begin{aligned}
 (\mathcal{E}_1, \mathcal{S}_1, \mathcal{N}_1) &\leftarrow \{x_{\text{Robert}}^{\text{Tue}} = -\} \cup \{\text{Demand}^{\text{Tue}} = 4\} \models \{x_{\text{Megan}}^{\text{Tue}} = x_{\text{Kevin}}^{\text{Tue}} = x_{\text{Amy}}^{\text{Tue}} = x_{\text{Jon}}^{\text{Tue}} = \text{D}\} \\
 (\mathcal{E}_2, \mathcal{S}_2, \mathcal{N}_2) &\leftarrow \{x_{\text{Kevin}}^{\text{Mon}} = -, x_{\text{Kevin}}^{\text{Tue}} = \text{D}\} \cup \{\text{MinConsOff} = 2\} \models \{x_{\text{Kevin}}^{\text{Sun}} = -\} \\
 (\mathcal{E}_3, \mathcal{S}_3, \mathcal{N}_3) &\leftarrow \{x_{\text{Kevin}}^{\text{Sun}} = -\} \cup \{\text{Demand}^{\text{Sun}} = 5\} \models \perp
 \end{aligned}$$

Explanation Construction Pipeline



Explanation Construction Pipeline



Greedy Enumeration. We search the constraint space by testing subsets S of increasing size ($|S| = 1, 2, 3, \dots$)

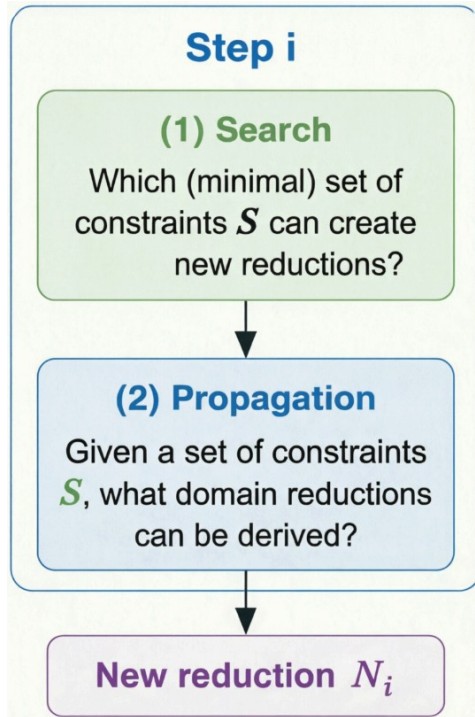
(!) Enumerating potentially large number of S

Standard Off-the-Shelf Propagation

$$N = \text{FullPropagate}(E \cup S) \setminus E$$

(!) Overhead of FullPropagation

Explanation Construction Pipeline



We proposed:

- **More efficient search algorithm** for S using **Divide-and-Conquer** approach
- **More efficient propagation** leveraging Multi-valued Decision Diagrams (MDDs)



Reframing Constraints

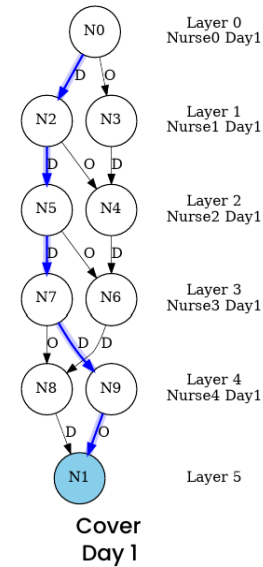
From Expressions to Pipes

Representation

- constraint = network of pipes

Need 4/5 nurses work on Day1

$$\sum_{i \in \text{Nurses}} (x_{\text{Nurse}_i, \text{Day1}} \leftarrow D) = 4$$



Multi-valued Decision Diagrams (MDDs)

Reframing Constraints

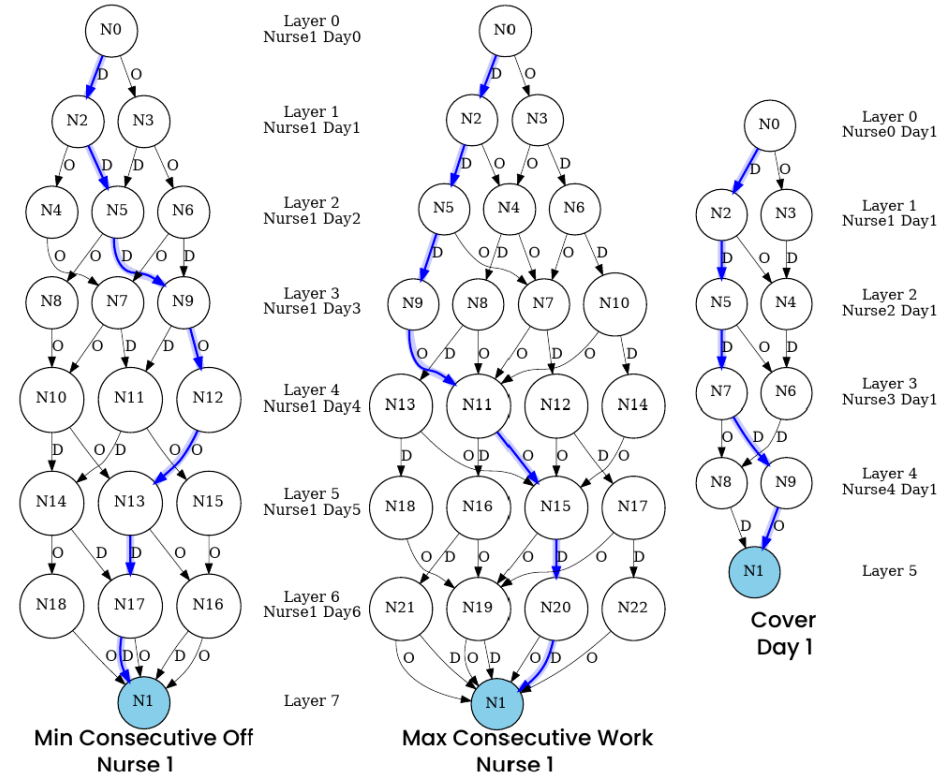
From Expressions to Pipes

Representation

- constraint = network of pipes

Decomposition

- problem = collection of MDDs

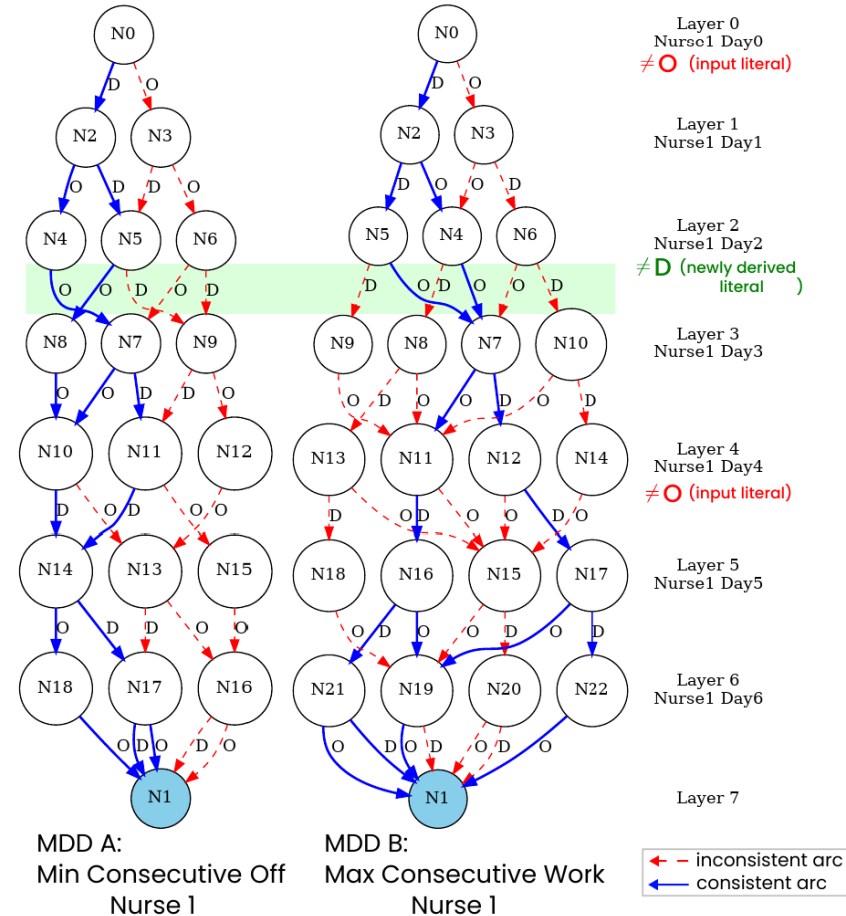


Multi-valued Decision Diagrams (MDDs)

Propagation

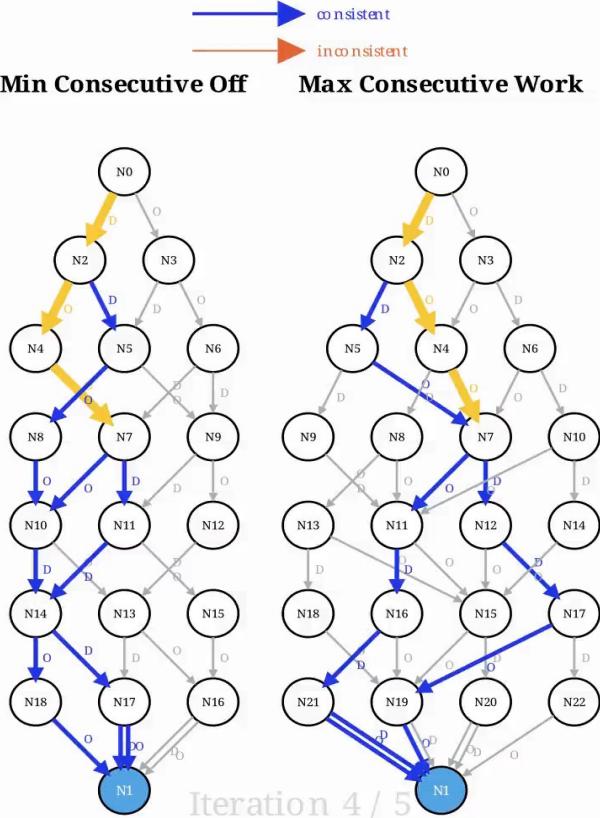
When Pipes Constrain Each Other

Propagation identifies **BLOCKAGE**
(*which pipes become infeasible*)
when a set of MDDs work together



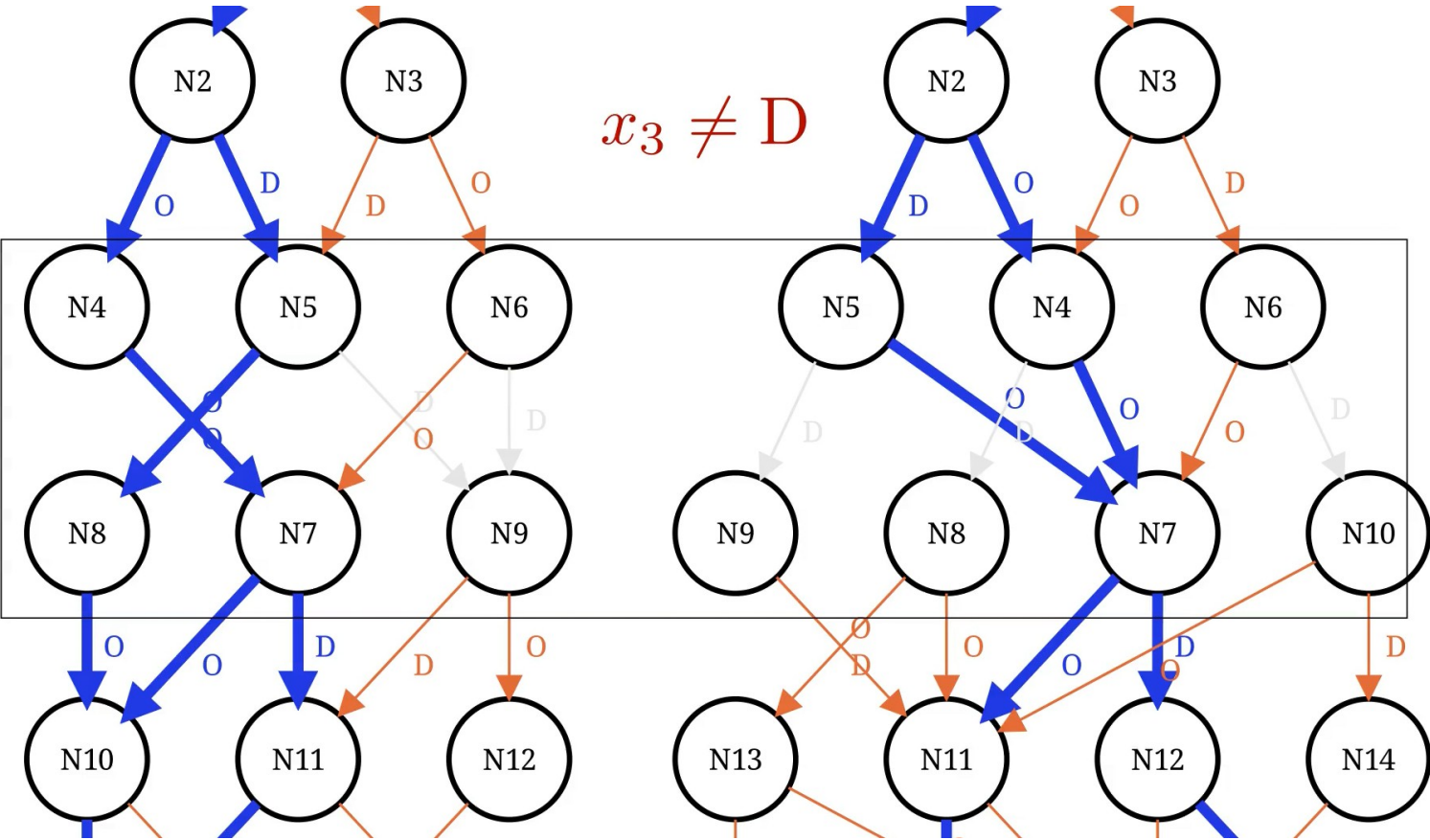
Propagation

When Blockage is Large Enough -> Domain Reduction



Propagation

When Blockage is Large Enough → Domain Reduction



Finding the Blockage: Network Flow Formulation

How do we compute which arcs are infeasible?

MILP Formulation

Find consistent longest paths through all MDDs

“consistent” = one variable one value

$$NF(\mathcal{E}, \mathcal{S}) : \max Z$$

max flows
to unsupported arcs

$$\text{s.t.} \quad \sum_{v \in D_x} y_{x,v} = 1 \quad \forall x \in \mathcal{X}_S$$

one variable,
one domain

$$\sum_{a \in \text{in}(u)} z_a^k = \sum_{a \in \text{out}(u)} z_a^k \quad \forall k \in \mathcal{S}, u \in V_k \setminus \{\mathbf{r}_k, \mathbf{t}_k\}$$

valid s-t flow constraint

$$\sum_{a \in A_k(x,v)} z_a^k = y_{x,v} \quad \forall k \in \mathcal{S}, x \in \mathcal{X}_k, v \in D_x$$

consistent constraint
linking z and y

$$y_{x,v} = 0 \quad \forall (x,v) \in \mathcal{E}$$

zero flow on
domain-reduced arcs

$$Z \leq \sum_{k \in \mathcal{S}} \sum_{a \in \overline{A}_k} z_a^k$$

obj bounded by
unsupported arc flows

$$z_a^k \in \{0, 1\} \quad \forall k \in \mathcal{S}, a \in A_k$$

$$y_{x,v} \in \{0, 1\} \quad \forall x \in \mathcal{X}_S, v \in D_x$$

Finding the Blockage: Network Flow Formulation

How do we compute which arcs are infeasible?

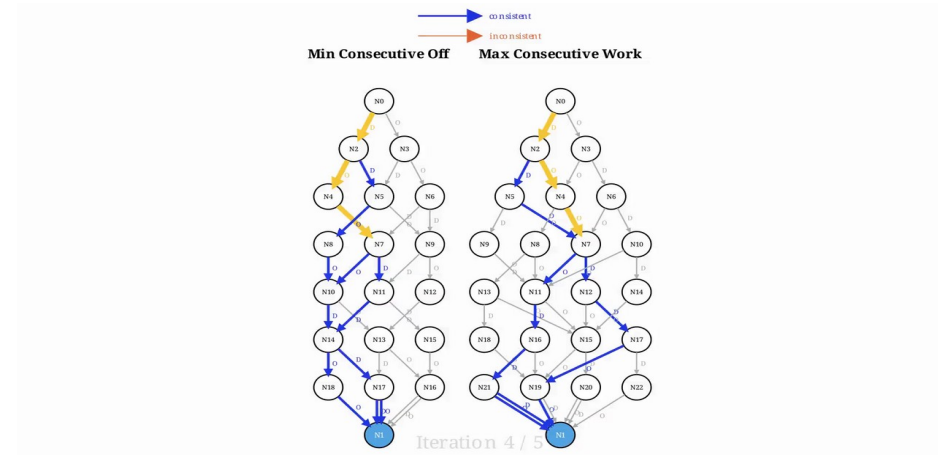
MILP Formulation

Find consistent longest paths through all MDDs

"consistent" = one variable one value

Iteratively solve **NF** until no more new pipe can be traversed

-> The remaining arcs are infeasible



Finding the Blockage: Network Flow Formulation

How do we compute which arcs are infeasible?

MILP Formulation

Find consistent longest paths through all MDDs

“consistent” = one variable one value

Iteratively solve **NF** until no more new pipe can be traversed

-> The remaining arcs are infeasible

$$NF(\mathcal{E}, \mathcal{S}) : \max Z$$

max flows
to unsupported arcs

$$\text{s.t.} \quad \sum_{v \in D_x} y_{x,v} = 1 \quad \forall x \in \mathcal{X}_S$$

one variable,
one domain

$$\sum_{a \in \text{in}(u)} z_a^k = \sum_{a \in \text{out}(u)} z_a^k \quad \forall k \in \mathcal{S}, u \in V_k \setminus \{\mathbf{r}_k, \mathbf{t}_k\}$$

valid s-t flow constraint

$$\sum_{a \in A_k(x,v)} z_a^k = y_{x,v} \quad \forall k \in \mathcal{S}, x \in \mathcal{X}_k, v \in D_x$$

consistent constraint
linking z and y

$$y_{x,v} = 0 \quad \forall (x,v) \in \mathcal{E}$$

zero flow on
domain-reduced arcs

$$Z \leq \sum_{k \in \mathcal{S}} \sum_{a \in \bar{A}_k} z_a^k$$

obj bounded by
unsupported arc flows

$$z_a^k \in \{0, 1\} \quad \forall k \in \mathcal{S}, a \in A_k$$

$$y_{x,v} \in \{0, 1\} \quad \forall x \in \mathcal{X}_S, v \in D_x$$

NP-Hard

Finding the Blockage: Network Flow Formulation

How do we compute which arcs are infeasible?

Lazy Cuts

Repeat

solve NF

extract flow

mark visited pipes

update obj function



$$Z \leq \sum_{k \in \mathcal{S}} \sum_{a \in \bar{A}_k} z_a^k$$

dynamic update
objective function

$$\sum_{k \in \mathcal{S}} \sum_{a \in \bar{A}_k} z_a^k \geq 1$$

force flows through
unvisited pipes

***“give me flows that
traverse other unvisited pipes”***

complexity: $\mathcal{O}(|\text{Arcs}|)$

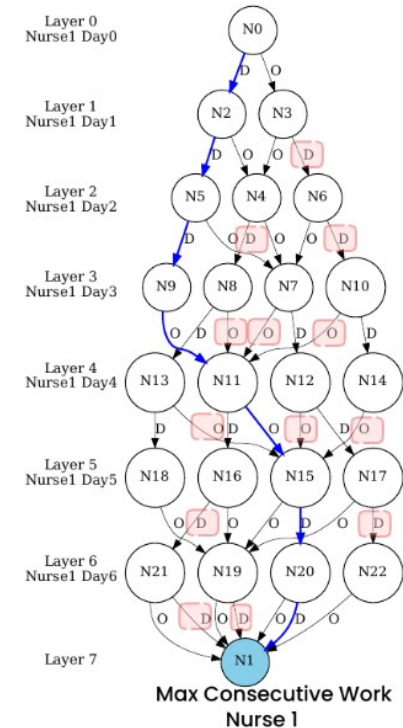
Finding the Blockage: Network Flow Formulation

Idea for tighter cuts

- Core question: "Has any variable-value assignment (domain) become infeasible?"
 -> only requires **domain-level consistency**
- If a single valid path uses an assignment ($x=v$), that domain is officially "supported."



lift lazy cuts from **arc variables** to
global domain variables



Finding the Blockage: Network Flow Formulation

Idea for tighter cuts

dynamic update
objective function

$$Z \leq \sum_{k \in \mathcal{S}} \sum_{a \in \bar{A}_k} z_a^k$$

force flows through
unvisited pipes

$$\sum_{k \in \mathcal{S}} \sum_{a \in \bar{A}_k} z_a^k \geq 1$$

Lazy Cuts

*“give me flows that
traverse other unvisited pipes”*

complexity: $\mathbf{O}(|\text{Arcs}|)$

$$Z \leq \sum_{(x,v) \in \bar{\mathcal{D}}} y_{x,v}$$

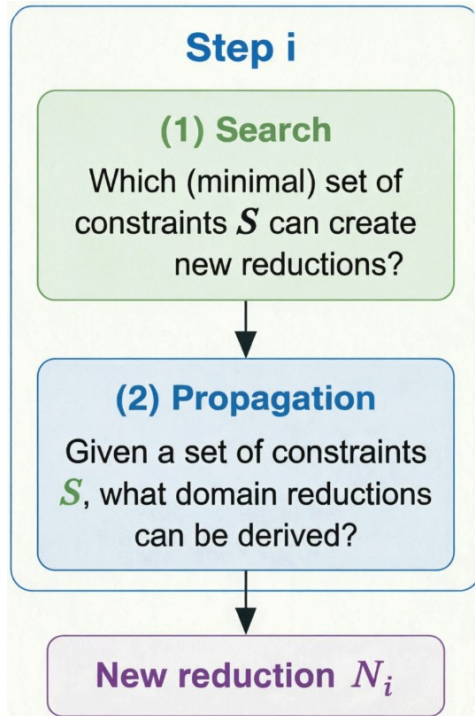
$$\sum_{(x,v) \in \bar{\mathcal{D}}} y_{x,v} \geq 1$$

Better Lazy Cuts

*“give me flows that
traverse other unvisited pipes,
from unvisited domains”*

complexity: $\mathbf{O}(|\text{Domains}|)$

Explanation Construction Pipeline



We proposed:

- **More efficient search algorithm** for S using **Divide-and-Conquer** approach
- ~~More efficient propagation~~ leveraging
- ~~Multi-valued Decision Diagrams (MDDs)~~

The Search: Minimal Explanation Steps

“find a constraint set that can create new domain reductions?”

Grow-Based Search

Step 2 / 75

■ tested

Propagation

{2}

NOTHING



(Bottom-Up)

Suffer combinatorial explosion

→ Inefficient

The Search: Minimal Explanation Steps

“find a constraint set that can create new domain reductions?”

Grow-Based Search

Step 2 / 75

■ tested

Propagation {2} **NOTHING**



Divide & Conquer Search

Step 2 / 11

■ tested ■ excluded

Propagation {1,2,3,4,5,6} **NOTHING**



(Bottom-Up)

Suffer combinatorial explosion
→ Inefficient

(Top-Down)

Recursively split MDD sets, test which subset causes new domain reductions

The Search: Minimal Explanation Steps

Implementation Optimizations

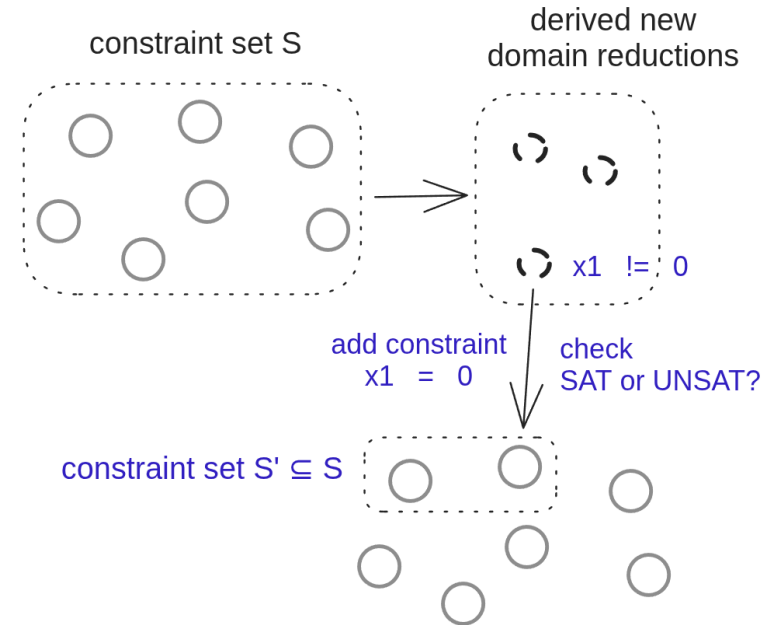
Component-based pruning & caching

- (Bleukx et al. 2023)

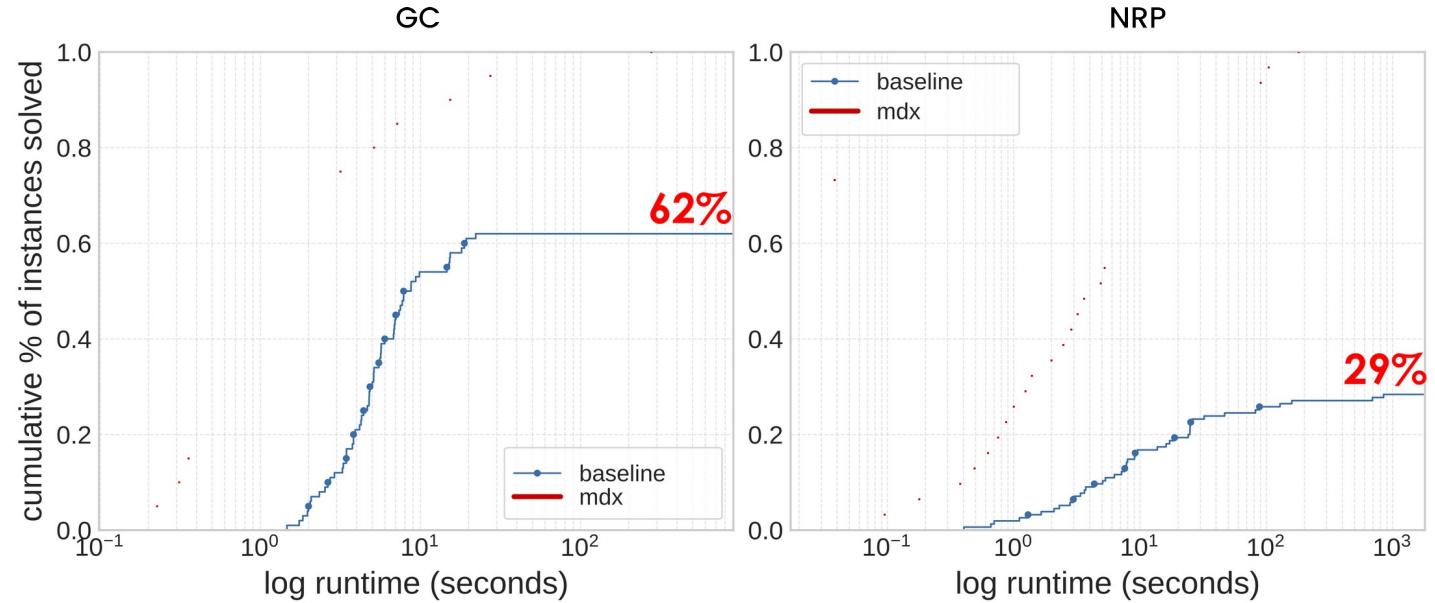
Modified MILP with contradiction for faster query

- leverage info of derived domain reductions to reduce optimization to feasibility problem

-> faster check in subsequent queries



Results

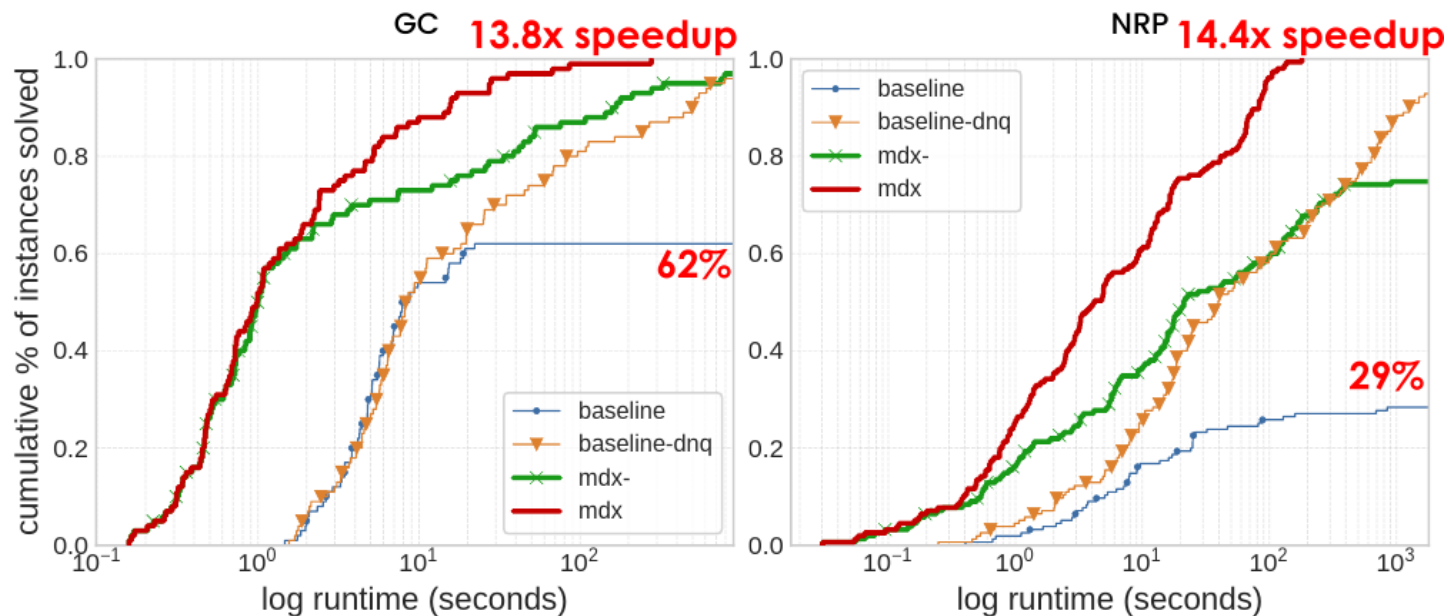


baseline(*)

Grow-based Enumeration + standard propagation

Results

**~14x
faster**



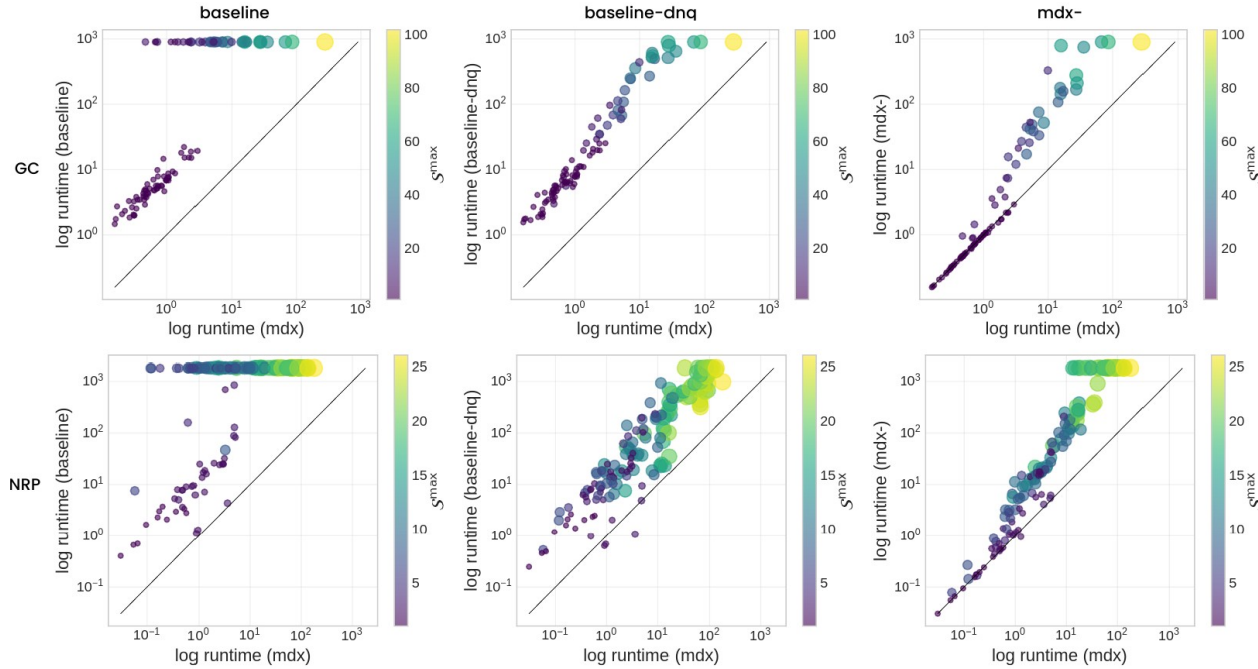
baseline(*)

baseline-dnq

mdx-

Grow-based Enumeration + standard propagation
Divide-and-conquer + standard propagation
MDD propagation by MILP with lazy cut

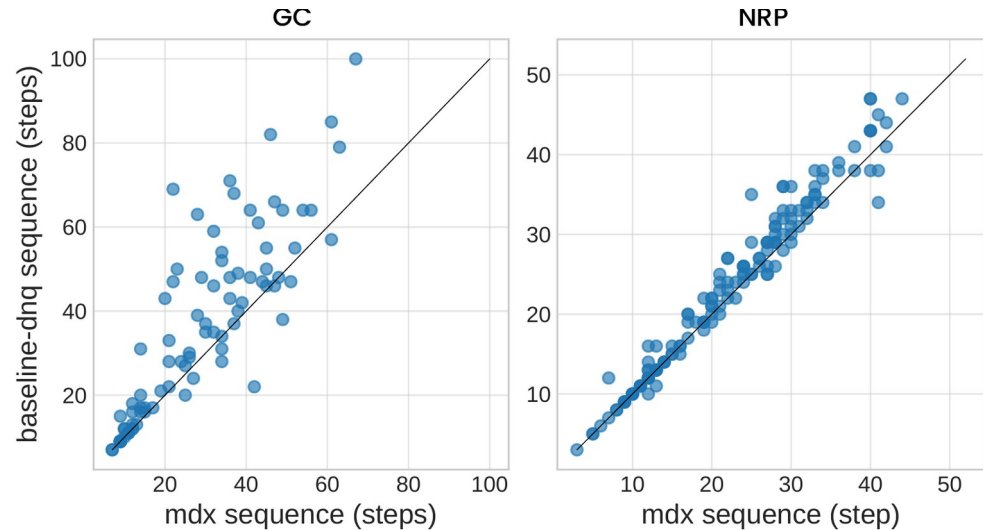
Results



- MDX routinely solves instances in <10 seconds that force baseline methods to hit the 15–30 minute timeout wall.
- Tighter Lazy Cuts and Fast Oracle Checks are essential to handle explanations with high complexity.

Metric: Sequence length (depth) used as proxy for explanation conciseness [Bleukx et al., 2023].

**Shorter or equal-length
explanation
in ~90% instances**



Conclusion: The Voice of the Flow

- We give the solver a **'voice'** through stepwise explanations, leveraging MDD representation, we use network **flow** formulation to accelerate the explanation generation.
- Bounded Search Space: A top-down divide-and-conquer strategy, coupled with domain-level lazy cuts and fast oracle checks.
- Achieves 100% solvability and magnitude speedups on benchmarks compared to state-of-the-art baselines

