Symbolic Abstraction with SMT Solvers

Yi Li[1] Aws Albarghouthi[1] Arie Gurfinkel[2] Zachary Kincaid[1] Marsha Chechik[1]

[1]Department of Computer Science
University of Toronto

[2]Software Engineering Institute
Carnegie Mellon University

December 17, 2013

Outline

Outline

UFO

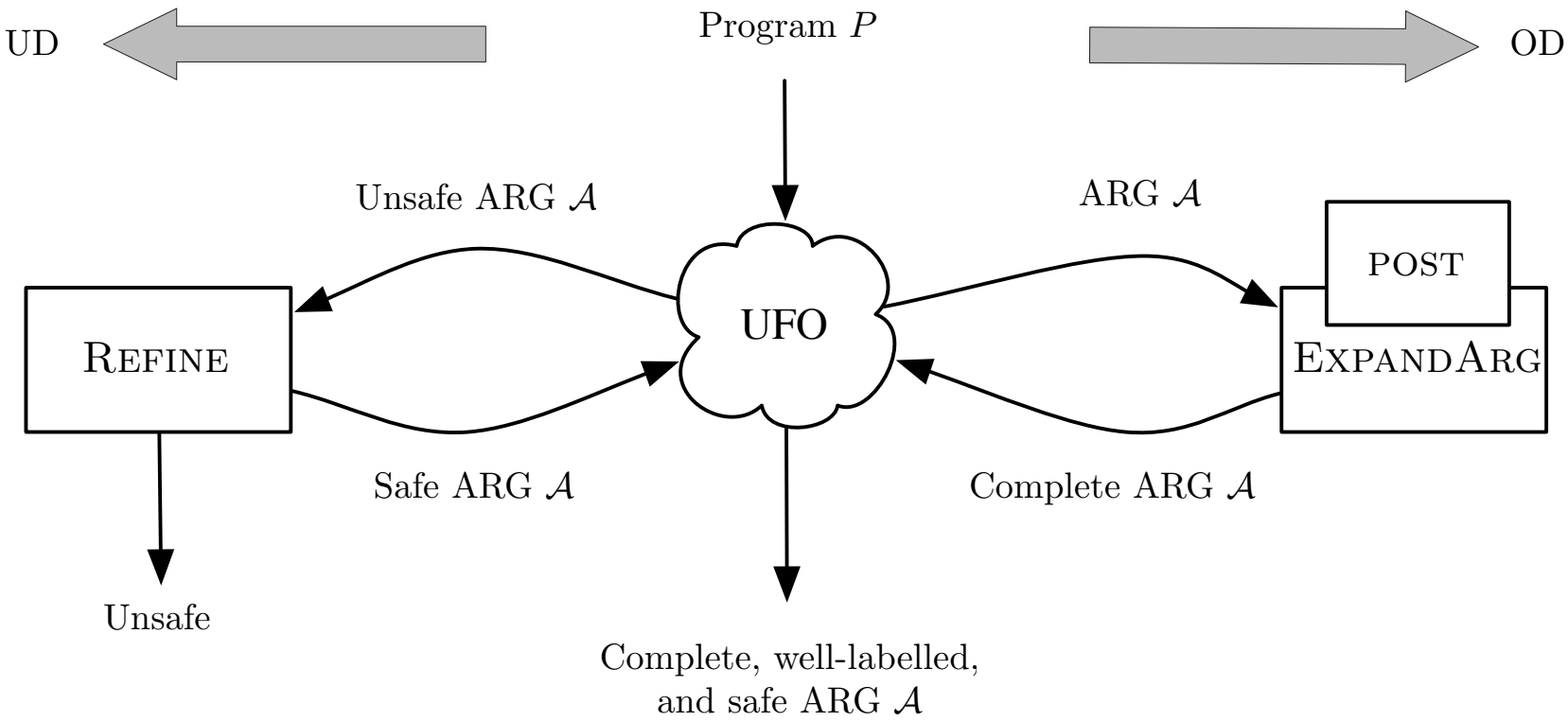


Figure: High level description of UFO.

Abstract Interpretation

```
1:y=0; x=0;

2:while (x<100) {
    x=x+2;
    y=y+2;
  }

3:assert (y>=0);
```

Intervals (BOX) domain

Abstract Interpretation

```
1:y=0; x=0;

2:while (x<100) {
    x=x+2;
    y=y+2;
  }

3:assert (y>=0);
```

Intervals (BOX) domain

1: $x = (-\infty, \infty), y = (-\infty, \infty)$

Abstract Interpretation

```
1:y=0; x=0;

2:while (x<100) {
    x=x+2;
    y=y+2;
  }

3:assert (y>=0);
```

Intervals (BOX) domain

1: $x = (-\infty, \infty), y = (-\infty, \infty)$

2: $x = [0, 0], y = [0, 0]$

Abstract Interpretation

```
1:y=0; x=0;

2:while (x<100) {
    x=x+2;
    y=y+2;
  }

3:assert (y>=0);
```

Intervals (BOX) domain

1: $x = (-\infty, \infty), y = (-\infty, \infty)$

2: $x = [0, 0], y = [0, 0]$

2: $x = [2, 2], y = [2, 2]$

Abstract Interpretation

```
1:y=0; x=0;

2:while (x<100) {
    x=x+2;
    y=y+2;
  }

3:assert (y>=0);
```

Intervals (BOX) domain

1: $x = (-\infty, \infty), y = (-\infty, \infty)$

2: $x = [0, 2], y = [0, 2]$

Abstract Interpretation

```
1:y=0; x=0;

2:while (x<100) {
    x=x+2;
    y=y+2;
  }

3:assert (y>=0);
```

Intervals (BOX) domain

1: $x = (-\infty, \infty), y = (-\infty, \infty)$

2: $x = [0, 2], y = [0, 2]$

Imprecision due to join

Abstract Interpretation

```
1:y=0; x=0;

2:while (x<100) {
    x=x+2;
    y=y+2;
  }

3:assert (y>=0);
```

Intervals (BOX) domain

1: $x = (-\infty, \infty), y = (-\infty, \infty)$

2: $x = [0, 2], y = [0, 2]$

Abstract Interpretation

```
1:y=0; x=0;

2:while (x<100) {
    x=x+2;
    y=y+2;
  }

3:assert (y>=0);
```

Intervals (BOX) domain

1: $x = (-\infty, \infty), y = (-\infty, \infty)$

2: $x = [0, 2], y = [0, 2]$

2: $x = [2, 4], y = [2, 4]$

Abstract Interpretation

```
1:y=0; x=0;

2:while (x<100) {
    x=x+2;
    y=y+2;
  }

3:assert (y>=0);
```

Intervals (BOX) domain

1: $x = (-\infty, \infty), y = (-\infty, \infty)$

2: $x = [0, \infty), y = [0, \infty)$

Abstract Interpretation

```
1:y=0; x=0;

2:while (x<100) {
    x=x+2;
    y=y+2;
  }

3:assert (y>=0);
```

Intervals (BOX) domain

1: $x = (-\infty, \infty), y = (-\infty, \infty)$

2: $x = [0, \infty), y = [0, \infty)$

Imprecision due to widening

Abstract Interpretation

	Intervals (BOX) domain
`1:y=0; x=0;`	1: $x = (-\infty, \infty), y = (-\infty, \infty)$

`2:while (x<100) {`
`  x=x+2;`
`  y=y+2;`
`}`
	2: $x = [0, \infty), y = [0, \infty)$

`3:assert (y>=0);`	3: $x = [100, \infty), y = [0, \infty)$

Imprecision due to widening

Sources of Imprecision

Join

- Avoids exploring exponentially many paths

Sources of Imprecision

Join

- Avoids exploring exponentially many paths

Widening

- Forces convergence

Sources of Imprecision

Join

- Avoids exploring exponentially many paths

Widening

- Forces convergence

Abstract post

- Imprecise interpretation of instructions, e.g. `y=x<<2`

Sources of Imprecision

Join

- Avoids exploring exponentially many paths

Widening

- Forces convergence

Abstract post

- Imprecise interpretation of instructions, e.g. `y=x<<2`

Abstract domain

- E.g., BOX cannot represent relation `y=x+2`

Symbolic Abstraction

Symbolic Abstraction

Symbolic Abstraction: $\widehat{\alpha}(\varphi)$

Itinerary

Monday:
✈ am. Baltra Airport
pm Highlands – Pit Craters (Santa Cruz)
Tuesday:
am. Egas Port – Salt Mines (Santiago)
pm. Bartolomé
Wednesday:
am. Dragon Hill (Santa Cruz
pm. North Seymour
Thursday:
✈ am. Baltra airport

A

Symbolic Abstraction: $\widehat{\alpha}(\varphi)$

Itinerary

Monday:
✈ am. Baltra Airport
pm Highlands – Pit Craters (Santa Cruz)
Tuesday:
am. Egas Port – Salt Mines (Santiago)
pm. Bartolomé
Wednesday:
am. Dragon Hill (Santa Cruz
pm. North Seymour
Thursday:
✈ am. Baltra airport

A


```
1:  x = y + 1;
2:  y = x - y;
3:  assert(y == 1);
```

Symbolic Abstraction: $\widehat{\alpha}(\varphi)$

Itinerary

Monday:
✈ am. Baltra Airport
pm Highlands – Pit Craters (Santa Cruz)
Tuesday:
am. Egas Port – Salt Mines (Santiago)
pm. Bartolomé
Wednesday:
am. Dragon Hill (Santa Cruz
pm. North Seymour
Thursday:
✈ am. Baltra airport

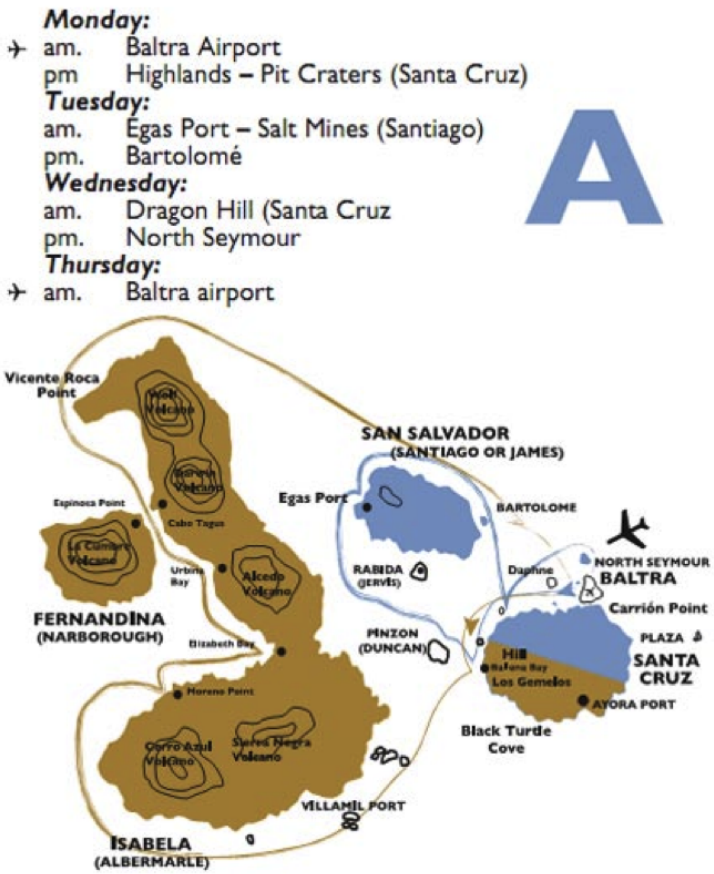
A


```
1:  x = y + 1;
2:  y = x - y;
3:  assert(y == 1);
```

$x : (-\infty, \infty) \quad y : (-\infty, \infty)$

Symbolic Abstraction: $\widehat{\alpha}(\varphi)$

Itinerary



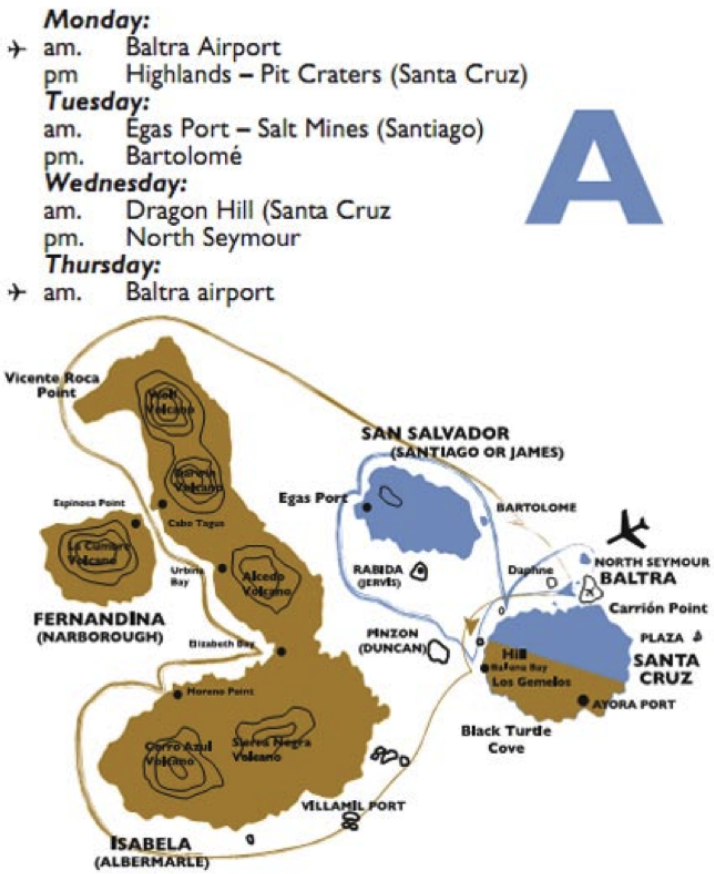

```
1:  x = y + 1;
2:  y = x - y;
3:  assert(y == 1);
```

$x : (-\infty, \infty) \quad y : (-\infty, \infty)$

$$\varphi \equiv x_0 = y_0 + 1 \wedge y_1 = x_0 - y_0$$

Symbolic Abstraction: $\widehat{\alpha}(\varphi)$

Itinerary




```
1:  x = y + 1;
2:  y = x - y;
3:  assert(y == 1);
```

$x : (-\infty, \infty) \quad y : (-\infty, \infty)$

$\varphi \equiv x_0 = y_0 + 1 \wedge y_1 = x_0 - y_0$

$y_1 : [1, 1]$

Symba

φ : QF_LRA

$\varphi \equiv x = 1$
$\varphi \equiv 1 \leqslant y \leqslant 3 \wedge (1 \leqslant x \leqslant 3 \vee x \geqslant 4)$

$\widehat{\alpha}(\varphi)$: TCM

$T = \{x\}$
$T = \{y, x + y\}$

SYMBA

atoms

φ : QF_LRA

$\varphi \equiv x = 1$
$\varphi \equiv 1 \leqslant y \leqslant 3 \wedge (1 \leqslant x \leqslant 3 \vee x \geqslant 4)$

$\widehat{\alpha}(\varphi)$: TCM

$T = \{x\}$
$T = \{y, x + y\}$

SYMBA

atoms

templates

φ : QF_LRA

$\varphi \equiv x = 1$

$\varphi \equiv 1 \leqslant y \leqslant 3 \wedge (1 \leqslant x \leqslant 3 \vee x \geqslant 4)$

$\widehat{\alpha}(\varphi)$: TCM

$T = \{x\}$

$T = \{y, x + y\}$

SYMBA

atoms

templates

φ : QF_LRA

$\varphi \equiv x = 1$

$\varphi \equiv 1 \leqslant y \leqslant 3 \wedge (1 \leqslant x \leqslant 3 \vee x \geqslant 4)$

$\widehat{\alpha}(\varphi)$: TCM

$T = \{x\}$

$T = \{y, x + y\}$

SYMBA

atoms

templates

φ : QF_LRA

$\varphi \equiv x = 1$
$\varphi \equiv 1 \leqslant y \leqslant 3 \wedge (1 \leqslant x \leqslant 3 \vee x \geqslant 4)$

SYMBA

$\widehat{\alpha}(\varphi)$: TCM

$T = \{x\}$
$T = \{y, x + y\}$

- Novel symbolic abstraction algorithm using SMT solvers.
- Maintain both under- and over-approximation of $\widehat{\alpha}(\varphi)$.
- Avoid imprecision by symbolically enumerating program paths.
- Parameterized by the TCM abstract domain that subsumes intervals, octagons and octahedra.

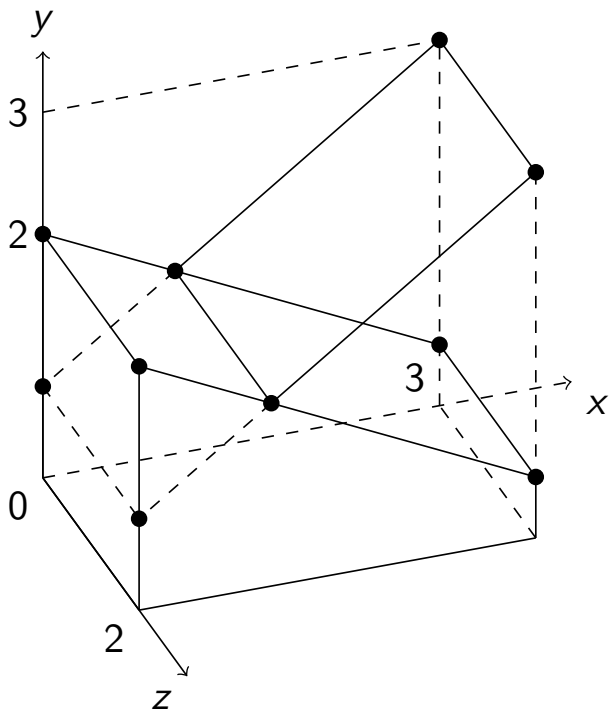
Outline

Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

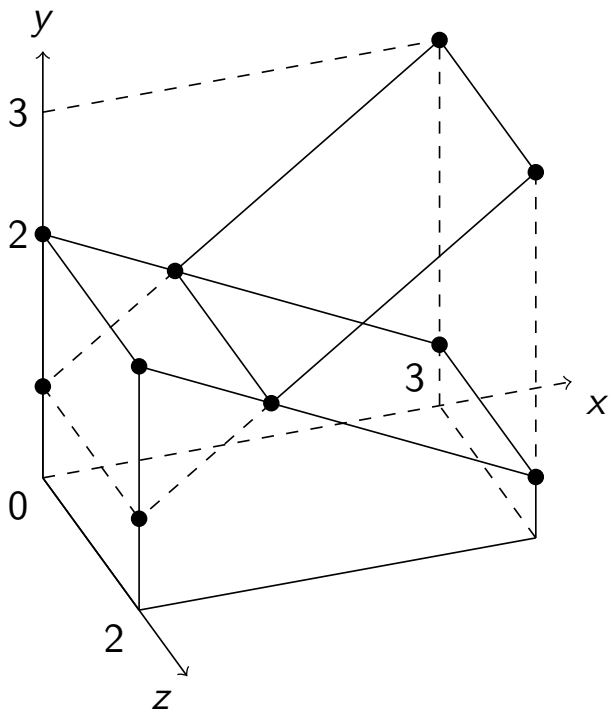
Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$



Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

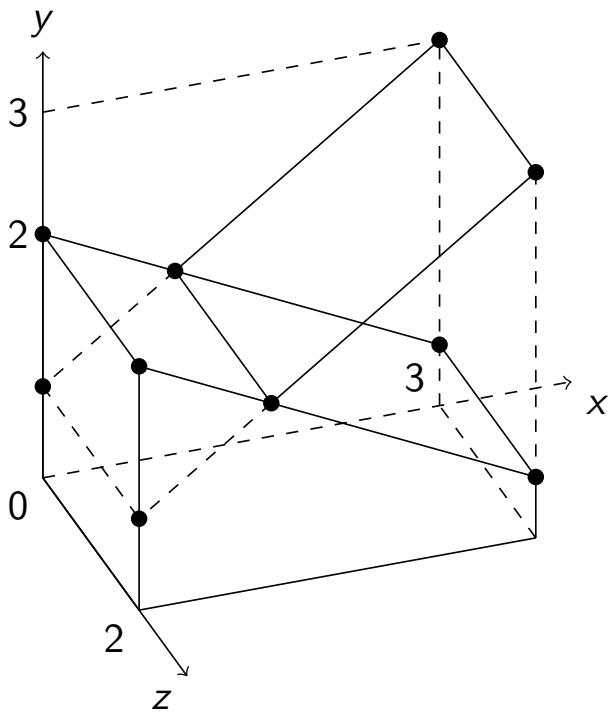


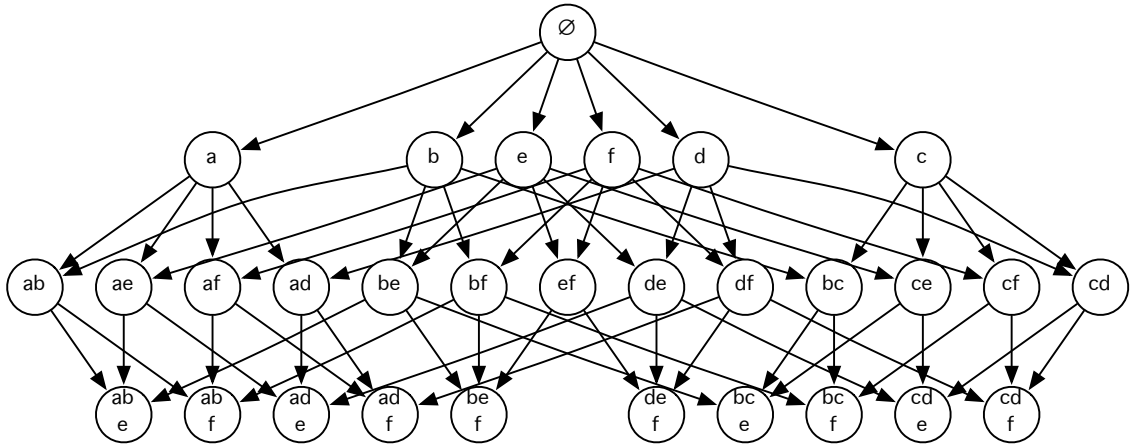
$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$

Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$

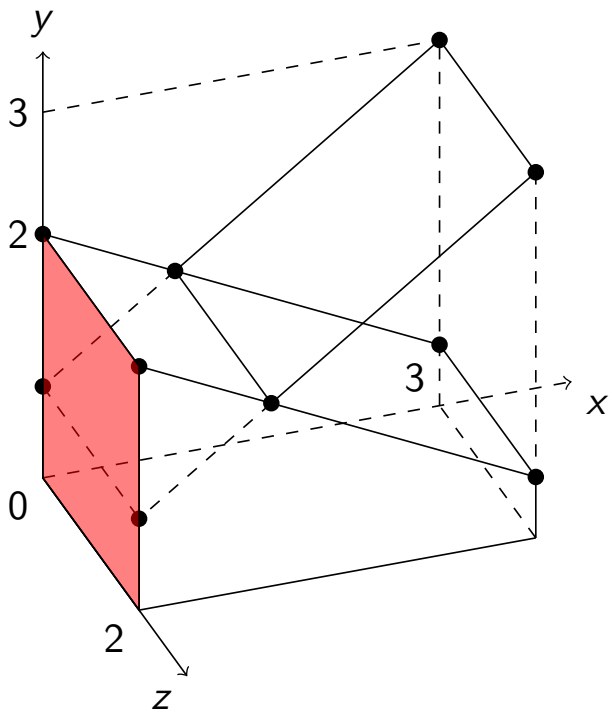


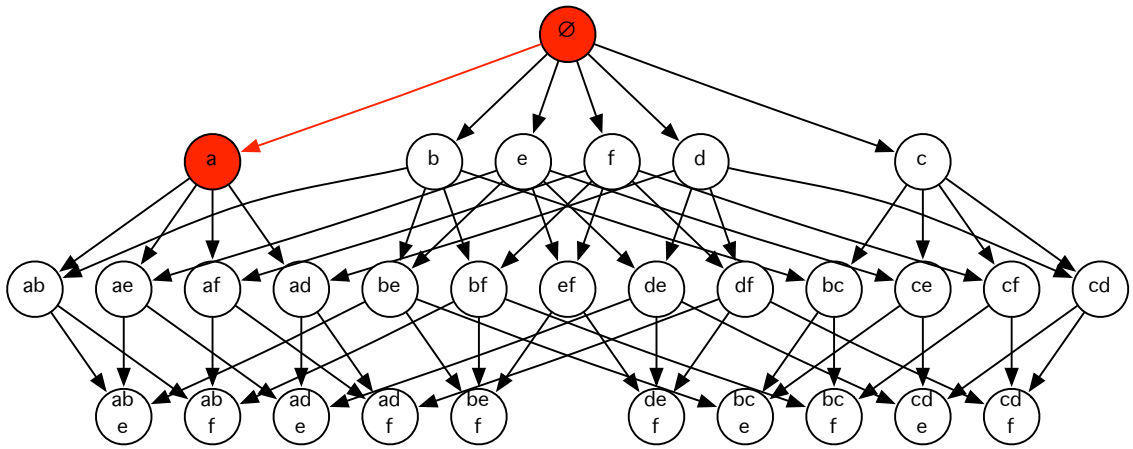


Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a &: \quad x = 0 \\ b &: \quad z = 2 \\ c &: \quad x = 3 \\ d &: \quad z = 0 \\ e &: \quad x + 2y = 4 \\ f &: \quad -3x + 4y = 3 \end{aligned}$$

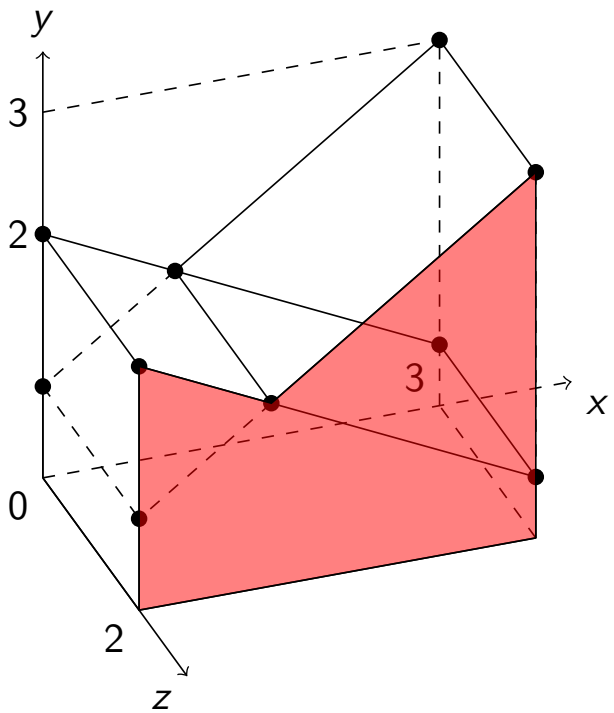


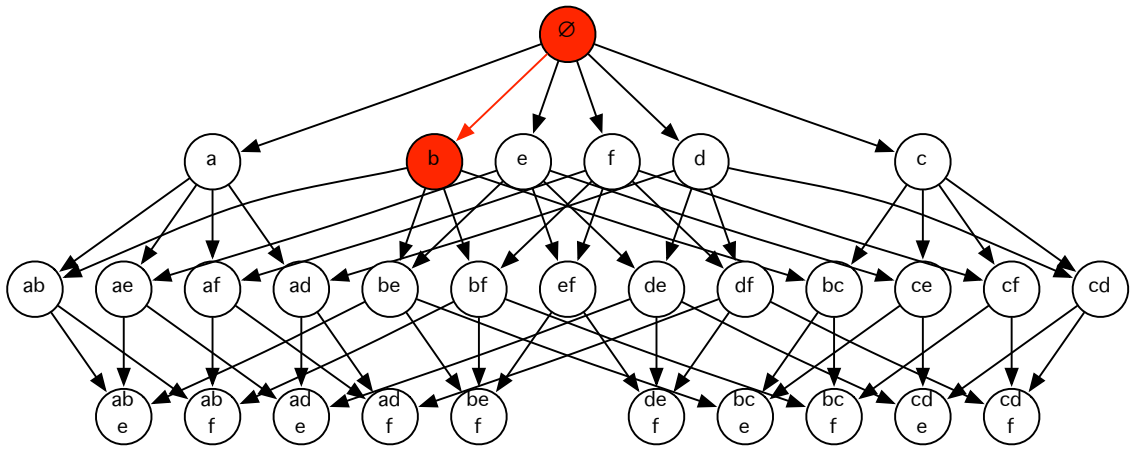


Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$

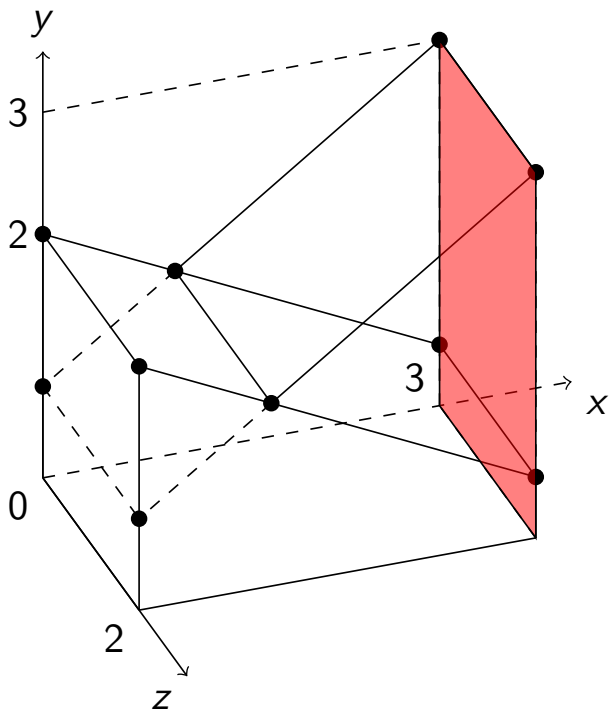


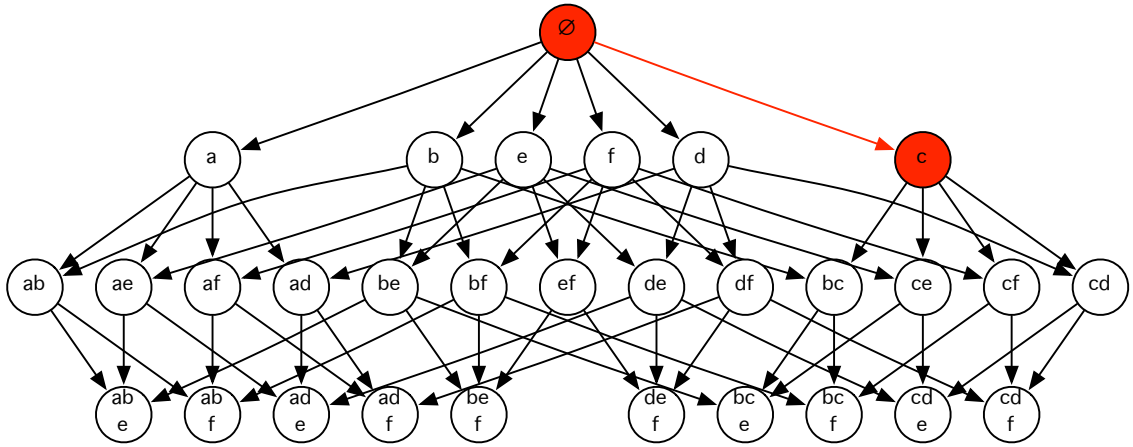


Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$

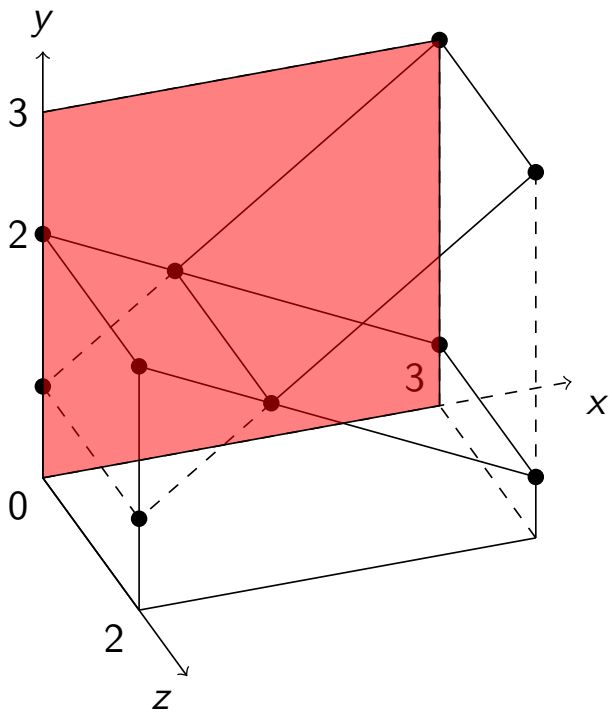


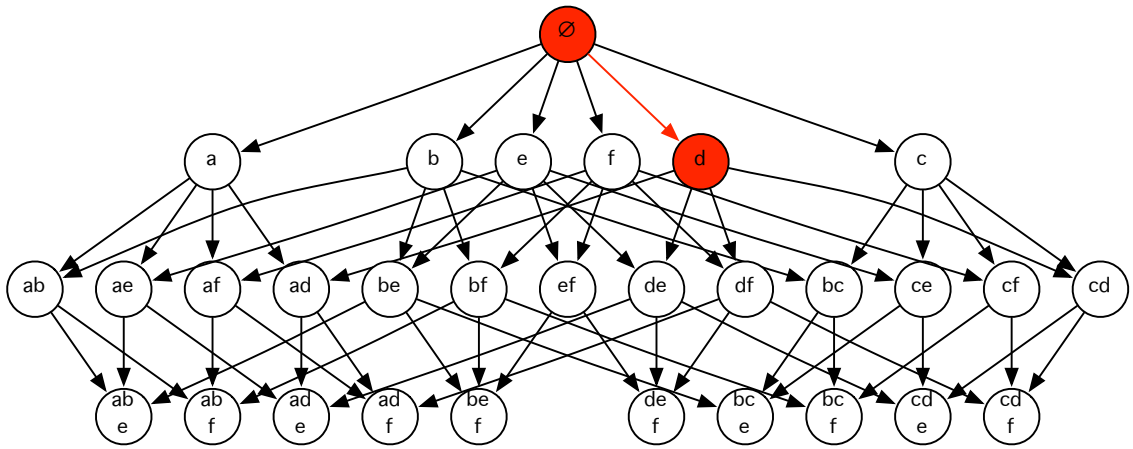


Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$

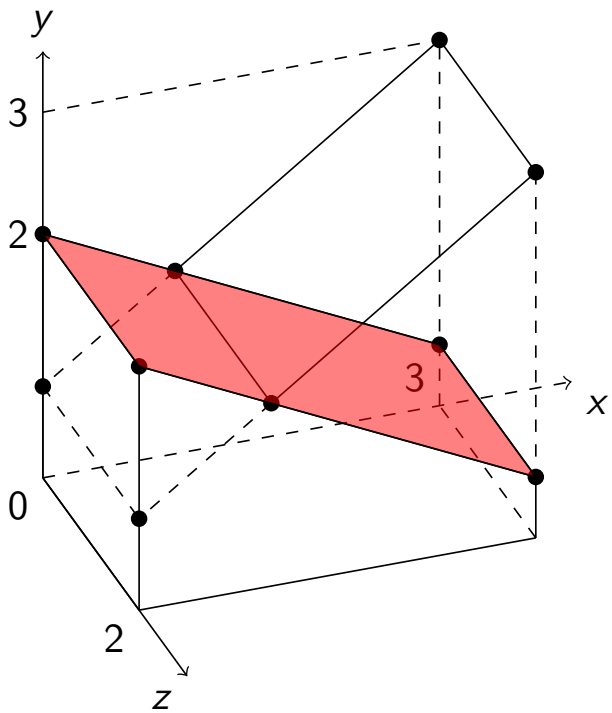


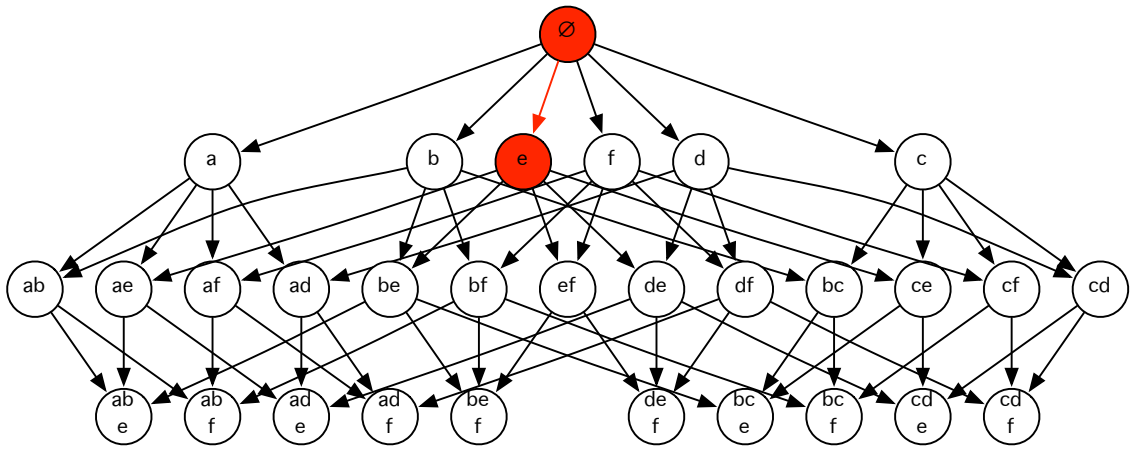


Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$

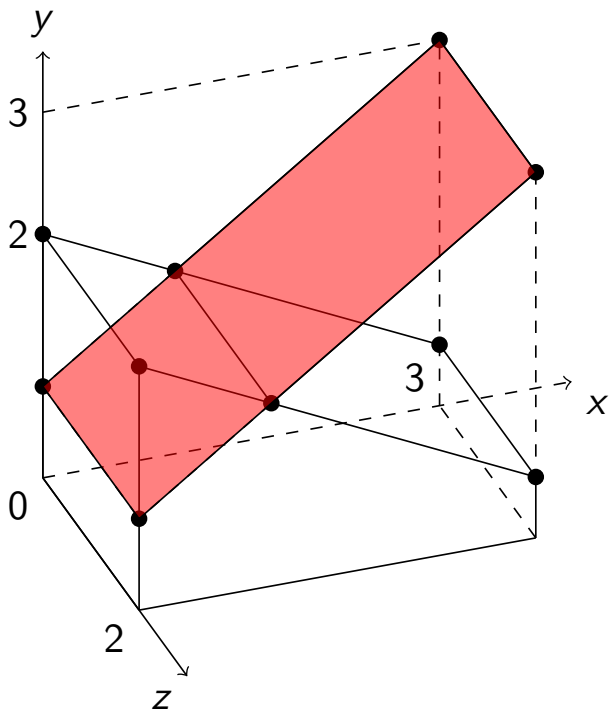


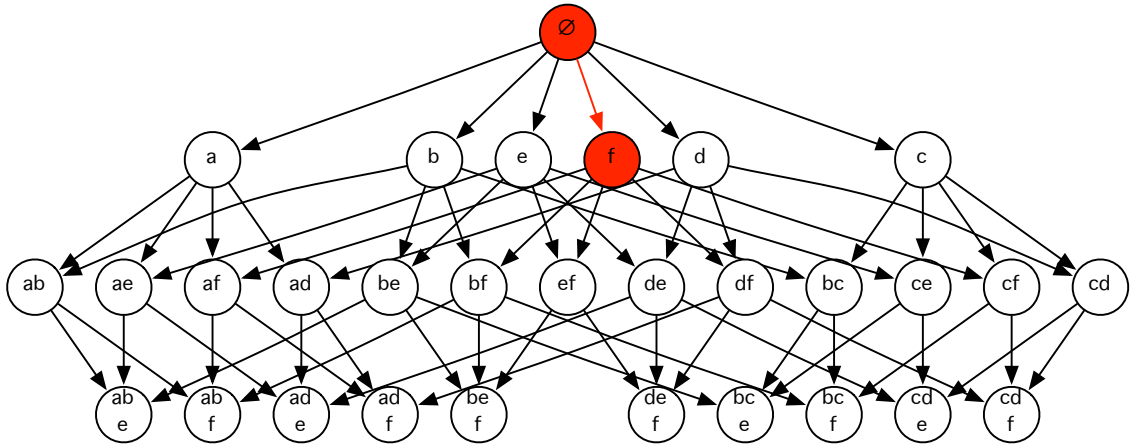


Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$

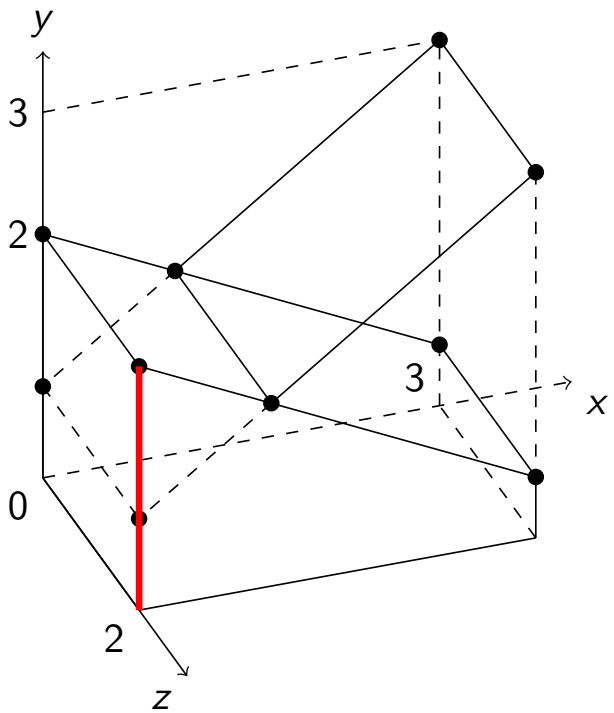


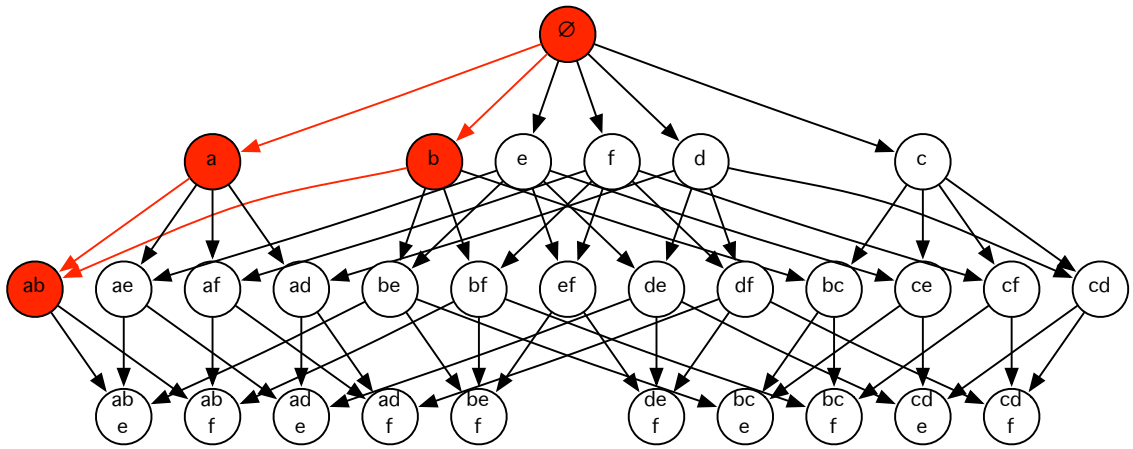


Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$

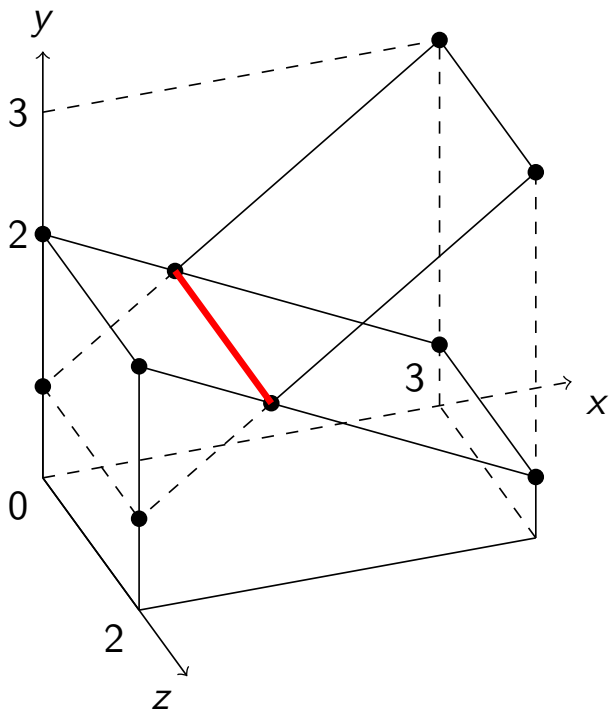


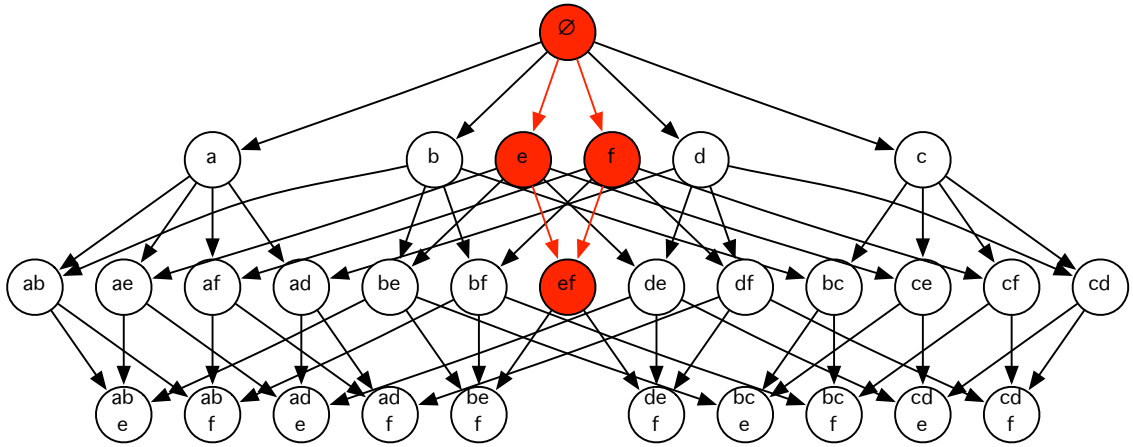


Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$

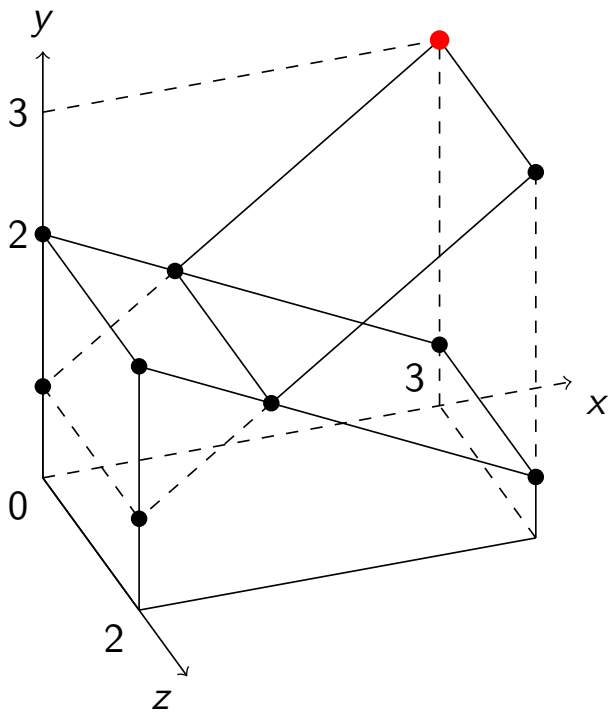


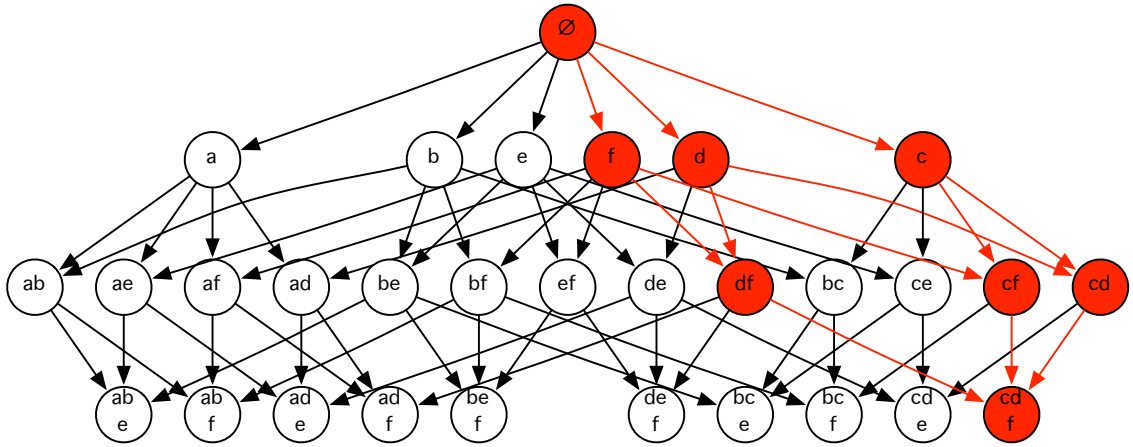


Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$

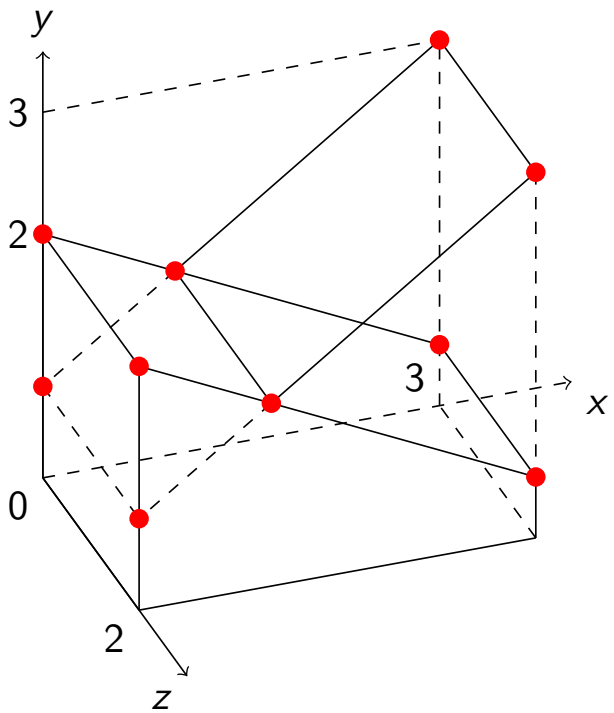


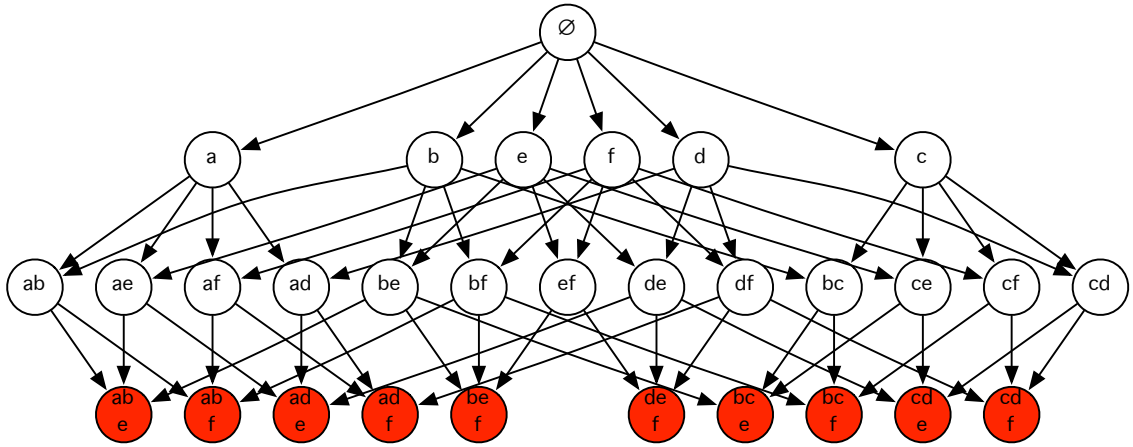


Equivalence Class: $[p]$

$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$

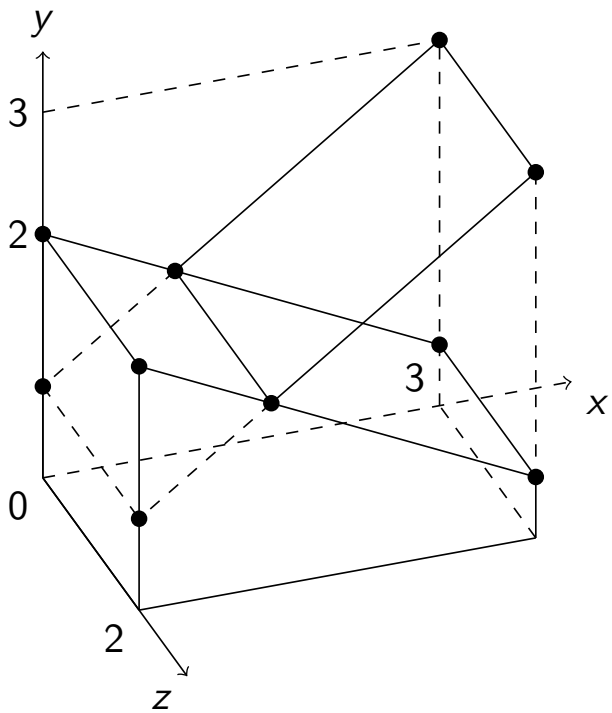


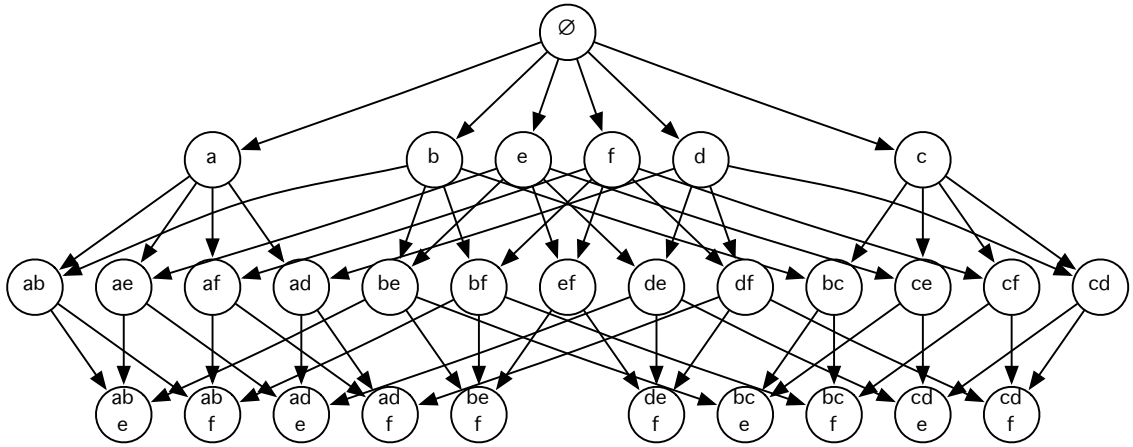


Equivalence Class: $[p]$

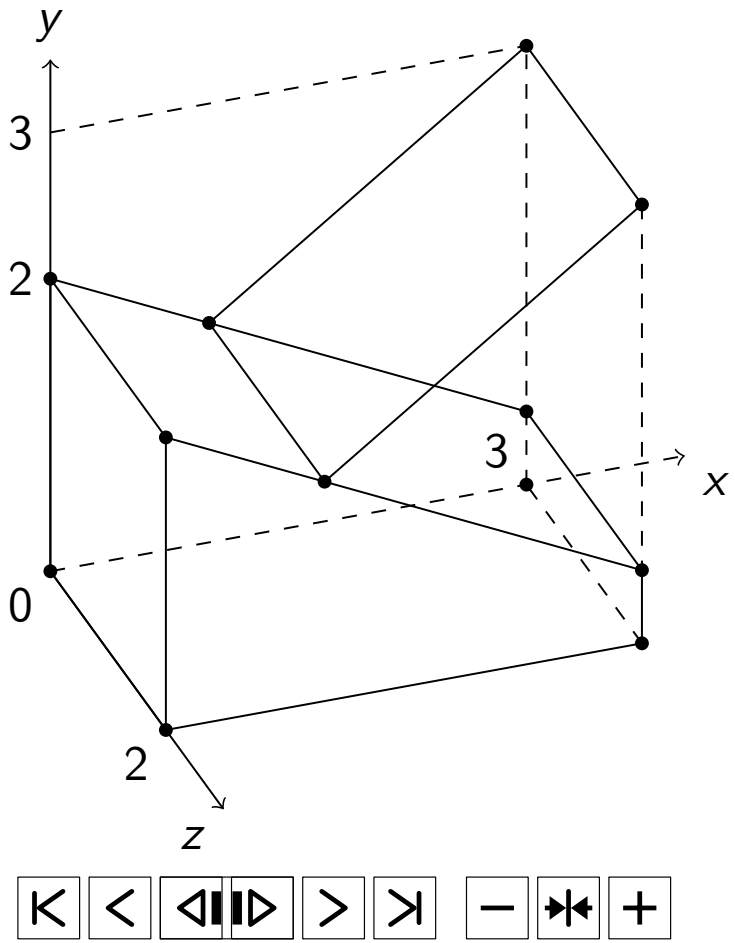
$$\varphi \equiv \quad 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3)$$

$$\begin{aligned} a: &\quad x = 0 \\ b: &\quad z = 2 \\ c: &\quad x = 3 \\ d: &\quad z = 0 \\ e: &\quad x + 2y = 4 \\ f: &\quad -3x + 4y = 3 \end{aligned}$$





A 3-dimensional Example



$$\varphi \equiv 0 \leqslant x \leqslant 3 \wedge 0 \leqslant z \leqslant 2 \wedge (2y \leqslant -x + 4 \vee 4y = 3x + 3) \quad T = \{y\}$$

Outline

Symba Formalized

$$\frac{}{\langle \emptyset, \bot, \top \rangle}\ \textsc{Init}$$

$$\frac{p \models \varphi \wedge \neg\widehat{\gamma}(U)}{\langle M, U, O\rangle \rightarrow \langle M \cup \{p\}, U \sqcup \widehat{\alpha}(p), O\rangle}\ \textsc{GlobalPush}$$

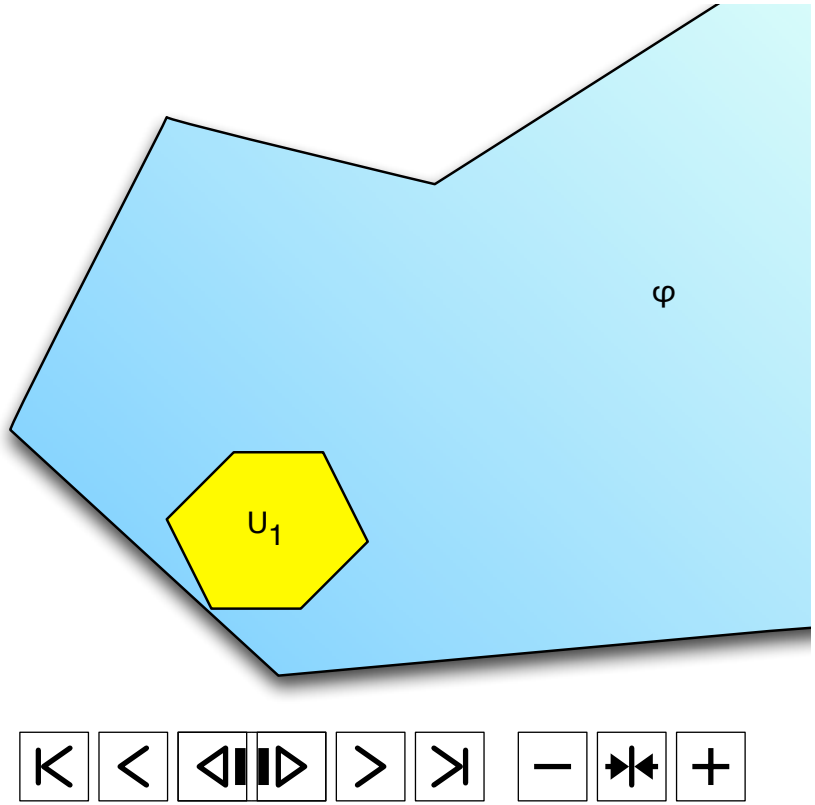
$$\frac{\begin{array}{c} U = (k_1, \ldots, k_n) \quad p_2 \models \varphi \quad [p_2] = [p_1] \quad t_i(p_1) < t_i(p_2) \\ \nexists p_3 \models \varphi \wedge t_i(p_2) \leqslant t_i(p_3) \wedge [p_2] \subset [p_3] \end{array}}{\langle M, U, O\rangle \rightarrow \langle M, U \sqcup (k_1, \ldots, k_{i-1}, \infty, k_{i+1}, \ldots, k_n), O\rangle}\ \textsc{Unbounded}(p_1 \in M, t_i \in T)$$

$$\frac{p_2, p_3 \models \varphi \quad t_i(p_1) < t_i(p_2) \leqslant t_i(p_3) \quad [p_1] = [p_2] \subset [p_3]}{\langle M, U, O\rangle \rightarrow \langle M \cup \{p_3\}, U \sqcup \widehat{\alpha}(p_3), O\rangle}\ \textsc{Unbounded-Fail}(p_1 \in M, t_i \in T)$$

$$\frac{O = (k_1, \ldots, k_n) \quad m = \max\{t_i(p') \mid p' \in M\} \quad \varphi \Rightarrow t_i \leqslant m}{\langle M, U, O\rangle \rightarrow \langle M, U, O \sqcap (k_1, \ldots, k_{i-1}, m, k_{i+1}, \ldots, k_n)\rangle}\ \textsc{Bounded}(t_i \in T)$$

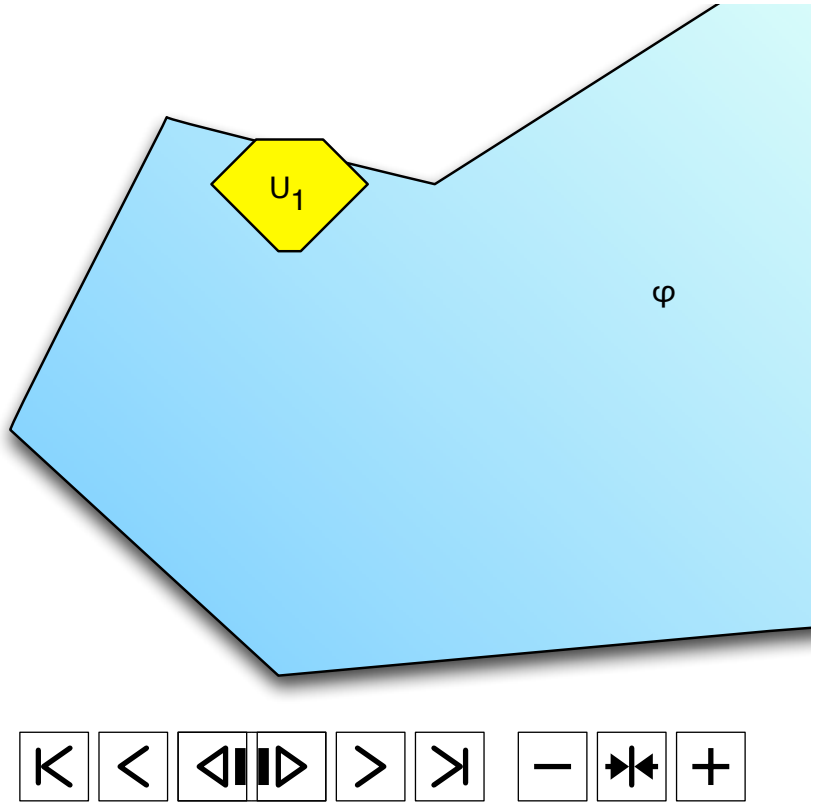
Figure: Inference rules used by Symba.

GlobalPush



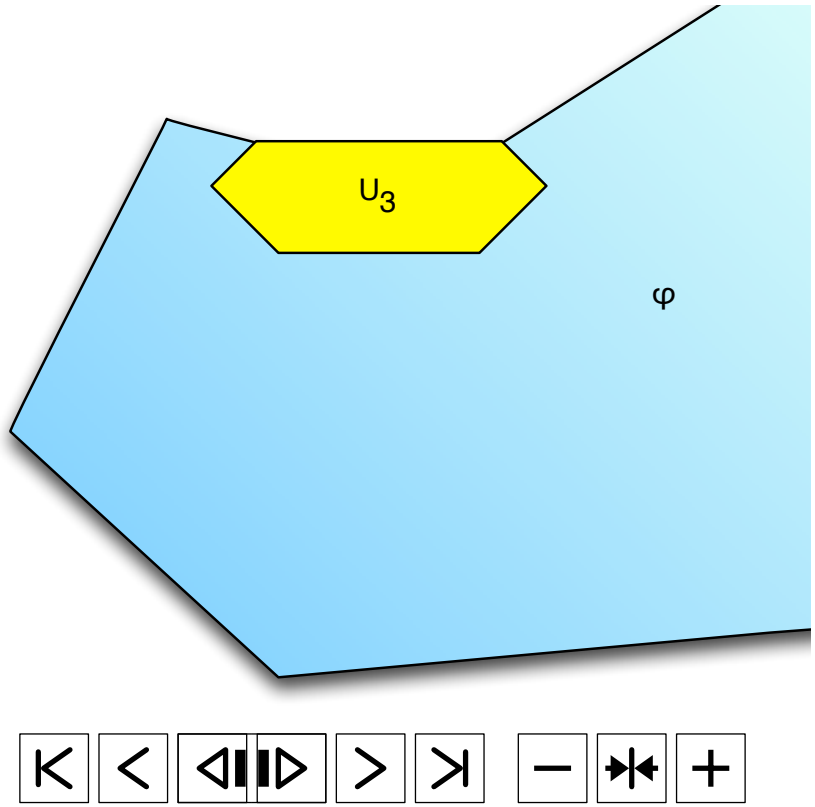
$$\frac{p \models \varphi \wedge \neg\widehat{\gamma}(U)}{\langle M, U, O\rangle \rightarrow \langle M \cup \{p\}, U \sqcup \widehat{\alpha}(p), O\rangle}\ \textsc{GlobalPush}$$

Unbounded-Fail



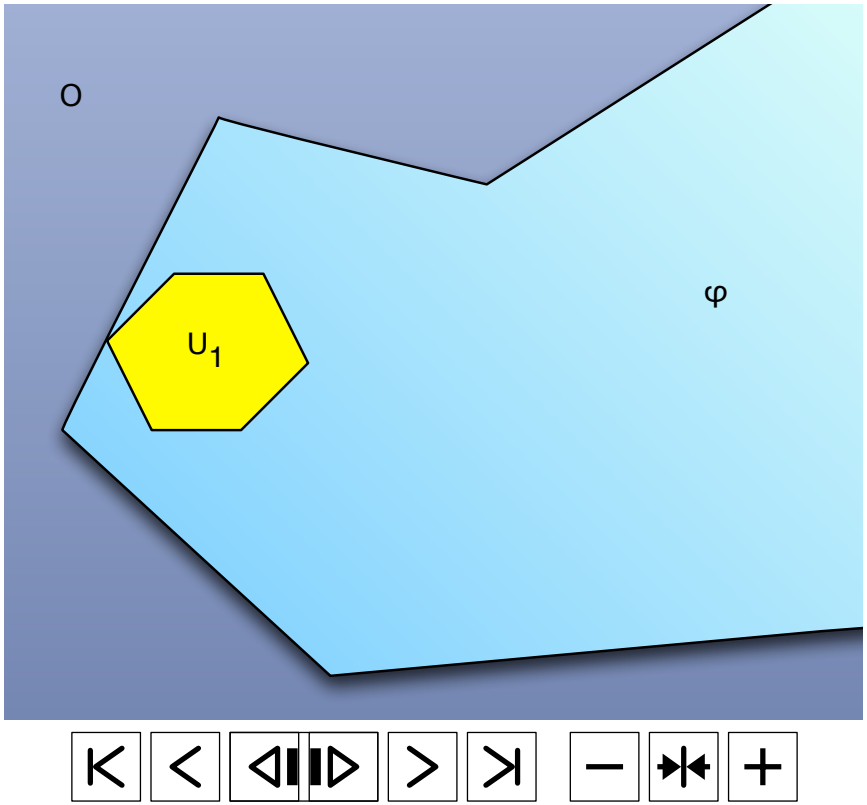
$$\frac{p_2, p_3 \models \varphi \quad t_i(p_1) < t_i(p_2) \leqslant t_i(p_3) \quad [p_1] = [p_2] \subset [p_3]}{\langle M, U, O\rangle \rightarrow \langle M \cup \{p_3\}, U \sqcup \widehat{\alpha}(p_3), O\rangle}\ \textsc{Unbounded-Fail}(p_1 \in M, t_i \in T)$$

Unbounded



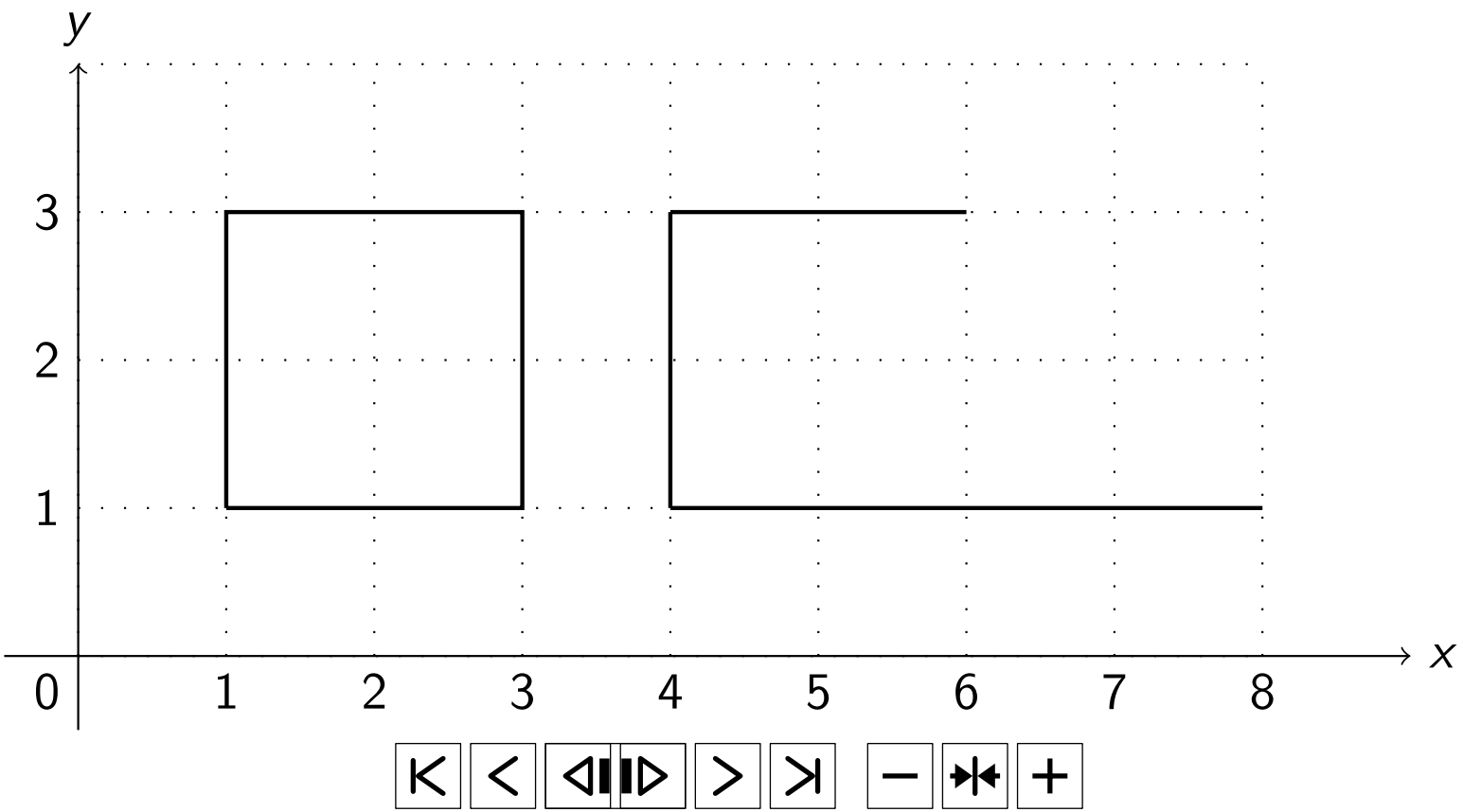
$$\frac{U = (k_1, \ldots, k_n) \quad p_2 \models \varphi \quad [p_2] = [p_1] \quad t_i(p_1) < t_i(p_2) \quad \nexists p_3 \models \varphi \wedge t_i(p_2) \leqslant t_i(p_3) \wedge [p_2] \subset [p_3]}{\langle M, U, O\rangle \rightarrow \langle M, U \sqcup (k_1, \ldots, k_{i-1}, \infty, k_{i+1}, \ldots, k_n), O\rangle}\ \textsc{Unbounded}(p_1 \in M, t_i \in T)$$

Bounded



$$\frac{O = (k_1, \ldots, k_n) \quad m = \max\{t_i(p') \mid p' \in M\} \quad \varphi \Rightarrow t_i \leqslant m}{\langle M, U, O\rangle \rightarrow \langle M, U, O \sqcap (k_1, \ldots, k_{i-1}, m, k_{i+1}, \ldots, k_n)\rangle}\ \textsc{Bounded}(t_i \in T)$$

A 2-dimensional Example



$$\varphi \equiv 1 \leqslant y \leqslant 3 \wedge (1 \leqslant x \leqslant 3 \vee x \geqslant 4) \quad T = \{y, x + y\}$$

UnboundedImpl

1: **function** UnboundImpl($c \in \mathcal{P}(\mathcal{E}(\varphi)), t_i \in T$)
2: Push()
3: Assert($t_i > U[t_i]$)
4: **if** UNSAT **then**
5: $O[t_i] \leftarrow U[t_i]$; Pop(); **return**
6: Assert($\bigwedge c$)
7: **if** UNSAT **then**
8: RemoveStronger($L(t), c$); Pop(); **return**
9: Assert($\bigvee(\mathcal{E}(\varphi) \setminus c)$)
10: **if** SAT **then** ▷ Unbounded-Fail
11: Pop(); **return** GetModel()
12: **else** ▷ Unbounded
13: $O[t_i] \leftarrow \infty$
14: Pop(); **return**

UnboundedImpl

1: **function** UnboundImpl($c \in \mathcal{P}(\mathcal{E}(\varphi)), t_i \in T$)
2: Push()
3: Assert($t_i > U[t_i]$)
4: **if** UNSAT **then**
5: $O[t_i] \leftarrow U[t_i]$; Pop(); **return**
6: Assert($\bigwedge c$)
7: **if** UNSAT **then**
8: RemoveStronger($L(t), c$); Pop(); **return**
9: Assert($\bigvee(\mathcal{E}(\varphi) \setminus c)$)
10: **if** SAT **then** ▷ Unbounded-Fail
11: Pop(); **return** GetModel()
12: **else** ▷ Unbounded
13: $O[t_i] \leftarrow \infty$
14: Pop(); **return**

UnboundedImpl

1: **function** UnboundImpl($c \in \mathcal{P}(\mathcal{E}(\varphi)), t_i \in T$)
2: Push()
3: Assert($t_i > U[t_i]$)
4: **if** UNSAT **then**
5: $O[t_i] \leftarrow U[t_i]$; Pop(); **return**
6: Assert($\bigwedge c$)
7: **if** UNSAT **then**
8: RemoveStronger($L(t), c$); Pop(); **return**
9: Assert($\bigvee(\mathcal{E}(\varphi) \setminus c)$)
10: **if** SAT **then** ▷ Unbounded-Fail
11: Pop(); **return** GetModel()
12: **else** ▷ Unbounded
13: $O[t_i] \leftarrow \infty$
14: Pop(); **return**

UnboundedImpl

```
1:  function UnboundImpl(c ∈ P(E(φ)), t_i ∈ T)
2:      Push()
3:      Assert(t_i > U[t_i])
4:      if UNSAT then
5:          O[t_i] ← U[t_i]; Pop(); return
6:      Assert(⋀ c)
7:      if UNSAT then
8:          RemoveStronger(L(t), c); Pop(); return
9:      Assert(⋁(E(φ) \ c)))
10:     if SAT then                                  ▷ Unbounded-Fail
11:         Pop(); return GetModel()
12:     else                                         ▷ Unbounded
13:         O[t_i] ← ∞
14:     Pop(); return
```

Soundness

Soundness of UNBOUNDED

Given a formula φ in logic $\mathcal{L}$ and a linear expression t over the variables of φ, then $\nexists k \in \mathbb{R} \cdot \varphi \Rightarrow t \leqslant k$ (i.e., t is unbounded) if and only if there exist $p_1, p_2 \models \varphi$ such that

1. $t(p_1) < t(p_2)$
2. $[p_1] = [p_2]$
3. $\nexists p_3 \models \varphi \cdot t(p_2) \leqslant t(p_3) \wedge [p_2] \subset [p_3]$

Termination

Fair Scheduling

A *fair scheduling* is an infinite sequence of actions $a_1, a_2, \ldots$, where

$$a_i \in \{\textsc{GlobalPush}, \textsc{Unbounded}, \textsc{Unbounded-Fail}\},$$

and the following conditions apply:

1. $\textsc{GlobalPush}$ appears infinitely often, and
2. if a point p is added to M along the execution sequence, then both $\textsc{Unbounded}(p, t)$ and $\textsc{Unbounded-Fail}(p, t)$ eventually appear.

Termination

Fair Scheduling

A *fair scheduling* is an infinite sequence of actions $a_1, a_2, \ldots$, where

$$a_i \in \{\textsc{GlobalPush}, \textsc{Unbounded}, \textsc{Unbounded-Fail}\},$$

and the following conditions apply:

1. $\textsc{GlobalPush}$ appears infinitely often, and
2. if a point p is added to M along the execution sequence, then both $\textsc{Unbounded}(p, t)$ and $\textsc{Unbounded-Fail}(p, t)$ eventually appear.

Termination

$\textsc{Symba}$ terminates after a finite number of actions in any fair execution.

Outline

Optimizations

Integer Rounding

$$\frac{U = (k_1, \ldots, k_n) \quad k_i \in \mathbb{R} \quad p \models \varphi \quad t_i(p) = \lceil k_i \rceil}{\langle M, U, O \rangle \rightarrow \langle M \cup \{p\}, U \sqcup \widehat{\alpha}(p), O \rangle} \, \mathrm{IR}(t_i \in T)$$

Optimizations

Integer Rounding

$$\frac{U = (k_1, \ldots, k_n) \quad k_i \in \mathbb{R} \quad p \models \varphi \quad t_i(p) = \lceil k_i \rceil}{\langle M, U, O \rangle \rightarrow \langle M \cup \{p\}, U \sqcup \widehat{\alpha}(p), O \rangle} \, \mathrm{IR}(t_i \in T)$$

Scheduling Policy

$$\textsc{GlobalPush}, \underbrace{\textsc{UnboundedImpl}, \ldots, \textsc{UnboundedImpl}}_{\textit{forcePush}}, \textsc{GlobalPush}$$

SMT-LIB2 Benchmarks

```
1: (declare-const x Real)
2: (declare-const y Real)
3: (declare-const k1 Real)
4: (declare-const k2 Real)
5: (assert (=>
6:  (and (and (<= 1.0 y) (<= y 3.0))
7:       (or (and (<= 1.0 x) (<= x 3.0))
8:           (>= x 4.0)))
9:  (and (< y k1) (< (+ x y) k2))))
```

Figure: Problem file for the 2-D example.

$$\varphi \equiv 1 \leqslant y \leqslant 3 \wedge (1 \leqslant x \leqslant 3 \vee x \geqslant 4) \quad T = \{y, x + y\}$$

Evaluation

- Evaluation of the effects of different optimizations on the efficiency of SYMBA.
- Comparison with other SMT-based symbolic abstraction algorithms.
- Precision comparison of invariants generated by SYMBA against traditional abstract transformers.

Evaluation

- Evaluation of the effects of different optimizations on the efficiency of Symba.
- Comparison with other SMT-based symbolic abstraction algorithms.
- Precision comparison of invariants generated by Symba against traditional abstract transformers.

SymbaIR(*forcePush*)	Symba with integer rounding. (*forcePush*) number of UnboundedImpl calls are forced before GlobalPush is called.
SymbaIROFF	SymbaIR(∞) without integer rounding.
DnfBound	Implemented a path-based algorithm described in [Monniaux & Gonnord, 2011] using APRON.
Z3Qelim	Applied Z3 quantifier elimination to compute the stongest post-condition.

SYMBA on SMT-LIB2 Benchmarks

CONFIGURATION	TIME(s)	# SMT	# SOLVE	# GP	# UB
SYMBAIR(1)	**1,707**	136,766	295	69,321	32,948
SYMBAIR(3)	560	74,217	295	16,138	30,861
SYMBAIR(8)	562	65,185	295	6,734	31,948
SYMBAIR(13)	**538**	65,019	295	5,502	32,603
SYMBAIR(∞)	569	69,785	295	**5,491**	**34,978**
SYMBAIROFF	708	72,680	295	**5,478**	**35,910**
DNFBOUND	1,562	208	**48**		
Z3QELIM	669	-	**39**		

Table: Overall results on 295 SMT-LIB2 benchmarks.

SYMBA on SMT-LIB2 Benchmarks Cont'd

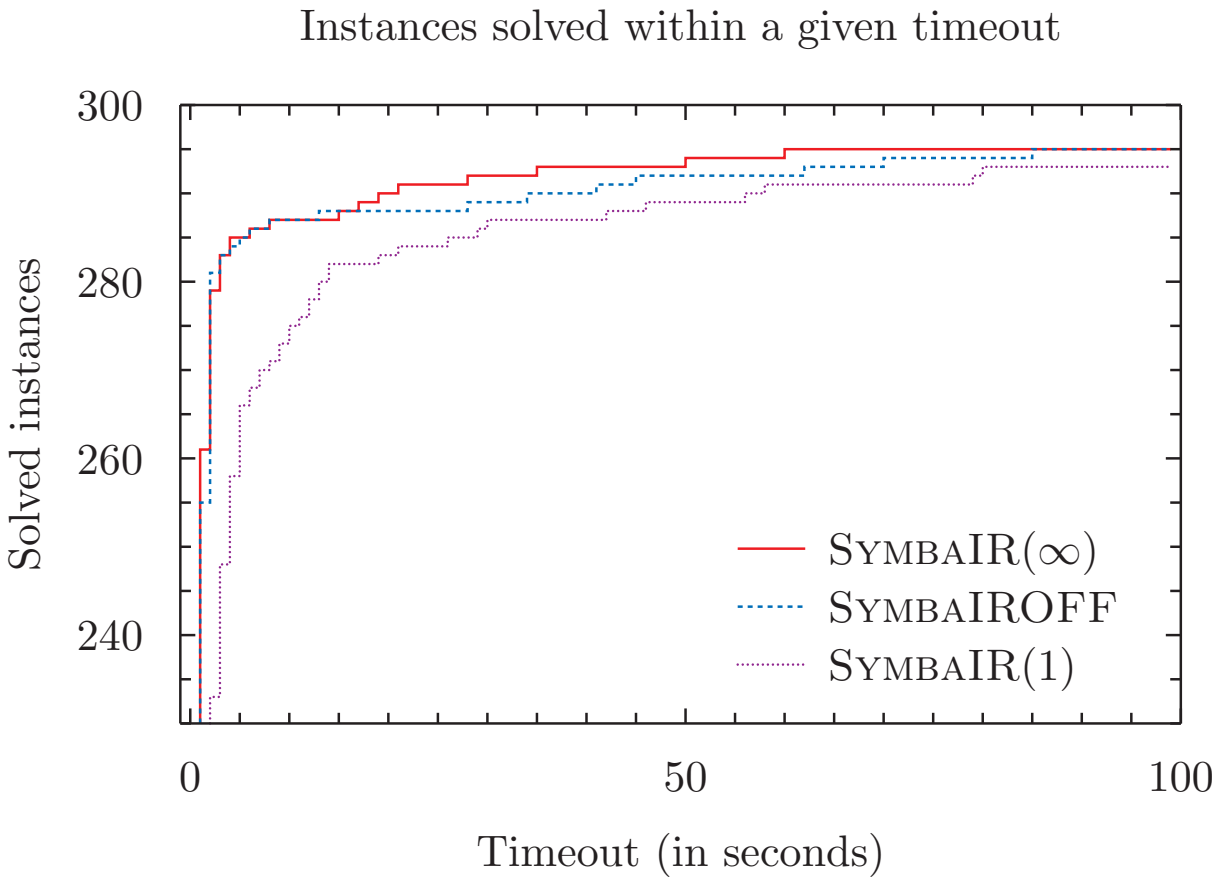


Figure: Number of benchmarks solved vs. timeout in seconds for several configurations of SYMBA.

SymbaIR(∞) vs. Intervals on Invariant Generation

AbsDom	# Completed	Time(s)	# Safety Proofs	Precision Gain(%)
SymbaIR(∞)	592	4,024	47	**73**
Intervals	604	121	0	0

Table: Overall results for loop invariant generation on 604 C benchmarks.

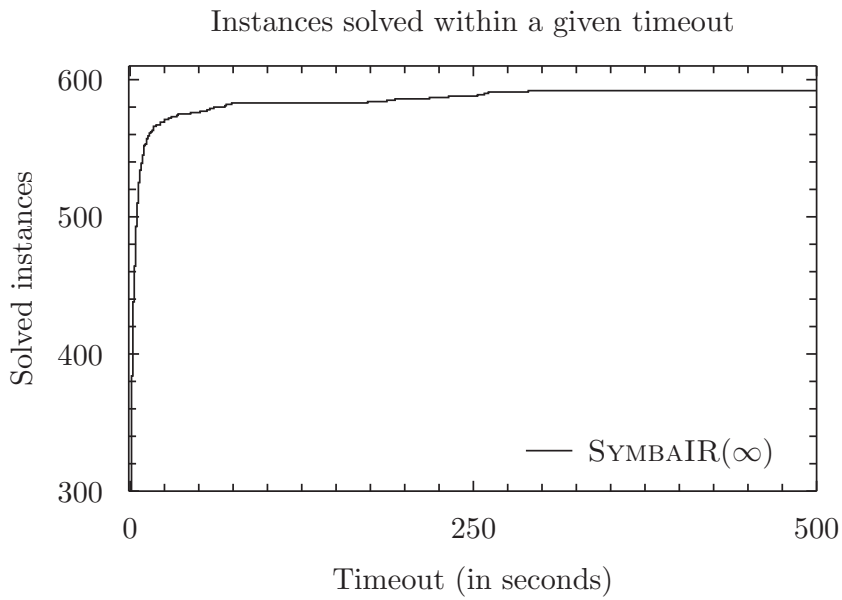


Figure: Number of C files analyzed using SymbaIR(∞) as the abstract transformer vs. timeout in seconds.

Observations

1. Our set of benchmarks is non-trivial.
2. SYMBA is more efficient than path-based algorithm DNFBOUND.
3. Integer rounding optimization and careful scheduling are important.
4. SYMBA brings a significant precision gain (73%).
5. It is fast in general.

Outline

Conclusion

1. Numerical invariant generation is important.
2. Abstract interpretation with numerical domains is not perfect.
   1. In-precise operations
   2. Trade-off between expressiveness and efficiency
3. SYMBA utilizes the power of SMT solvers and compute precise abstract post operator.
4. SYMBA enables invariant generation in the general TCM domain.
5. Experimental results show that SYMBA is efficient in practice compare to other symbolic abstraction approaches.

Future Work

1. Heuristics on scheduling policies.
2. Parallelize the implementation of SYMBA.
3. Extends to more SMT theories.

Thank You!