Interpolation goes on SAFARI

Francesco Alberti and Simone Fulvio Rollini

University of Lugano, Switzerland

July 14, 2013

Context: Software model checking

Software model checking:

- Given a program P and a (safety) property ϕ, does P exhibit an execution violating ϕ?

Context: Software model checking

Software model checking:

- Given a program P and a (safety) property ϕ, does P exhibit an execution violating ϕ?

- Transition-relation representation of input program
- Lazy Abstraction [HJMS02]
  - different degrees of precision for different parts of the program
- Interpolation-based refinement [HJMM04, McM06]

Lazy Abstraction with Interpolants

[McM06]

Original (Concrete) Program

```
1:   y = x;
2:   while ( x ≥ 1 ) {
3:         x = x - 1;
4:         y = y - 1;
5:   }
6:   if ( y ≥ 1 )
7:         ERROR;
```

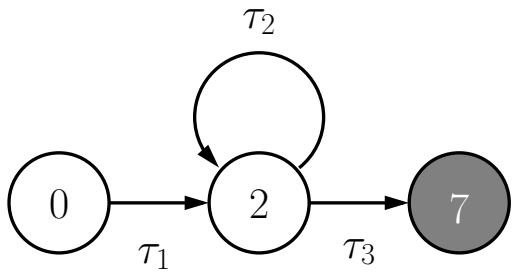
Lazy Abstraction with Interpolants

[McM06]

Original (Concrete) Program

```
1:  y = x;
2:  while ( x ≥ 1 ) {
3:        x = x - 1;
4:        y = y - 1;
5:  }
6:  if ( y ≥ 1 )
7:        ERROR;
```

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \left\{ \begin{array}{l} x' := x \\ y' := x \end{array} \right.$$

$$\tau_2\colon x \geq 1 \wedge \left\{ \begin{array}{l} x' := x - 1 \\ y' := y - 1 \end{array} \right.$$

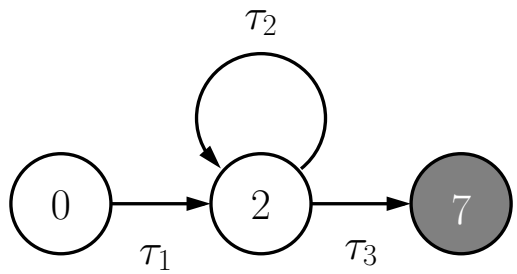
$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \left\{ \begin{array}{l} x' := x \\ y' := y \end{array} \right.$$

Lazy Abstraction with Interpolants

[McM06]

(Abstract) Program Unwinding

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon\ \top \wedge \left\{ \begin{array}{l} x' := x \\ y' := x \end{array} \right.$$

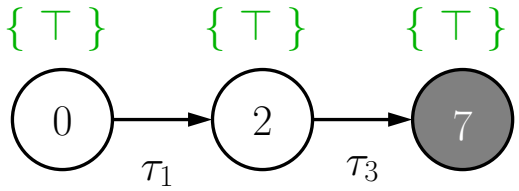
$$\tau_2\colon\ x \geq 1 \wedge \left\{ \begin{array}{l} x' := x - 1 \\ y' := y - 1 \end{array} \right.$$

$$\tau_3\colon\ x \leq 0 \wedge y \geq 1 \wedge \left\{ \begin{array}{l} x' := x \\ y' := y \end{array} \right.$$

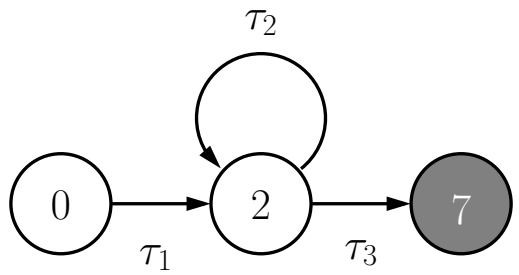
Lazy Abstraction with Interpolants

[McM06]

(Abstract) Program Unwinding



Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon\ \top \wedge \begin{cases} x' := x \\ y' := x \end{cases}$$

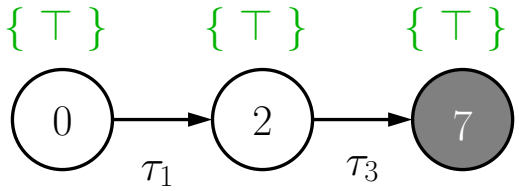
$$\tau_2\colon\ x \geq 1 \wedge \begin{cases} x' := x - 1 \\ y' := y - 1 \end{cases}$$

$$\tau_3\colon\ x \leq 0 \wedge y \geq 1 \wedge \begin{cases} x' := x \\ y' := y \end{cases}$$

Lazy Abstraction with Interpolants

[McM06]

(Abstract) Program Unwinding



true

$x_1 = x_0$

$y_1 = x_0$

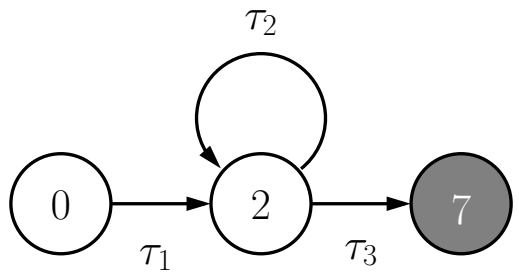
$x_1 \leq 0$

$y_1 \geq 1$

$x_2 = x_1$

$y_2 = y_1$

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \begin{cases} x' := x \\ y' := x \end{cases}$$

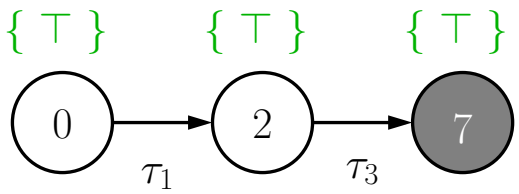
$$\tau_2\colon x \geq 1 \wedge \begin{cases} x' := x - 1 \\ y' := y - 1 \end{cases}$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \begin{cases} x' := x \\ y' := y \end{cases}$$

Lazy Abstraction with Interpolants

[McM06]

(Abstract) Program Unwinding



$\{ \top \}$

true

$x_1 = x_0$

$y_1 = x_0$

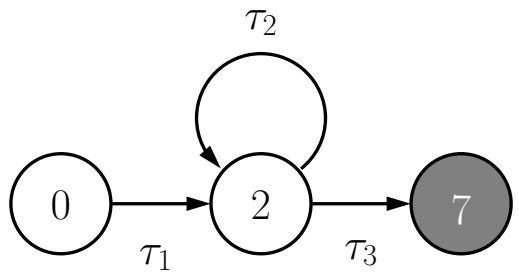
$x_1 \leq 0$

$y_1 \geq 1$

$x_2 = x_1$

$y_2 = y_1$

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \left\{ \begin{array}{l} x' := x \\ y' := x \end{array} \right.$$

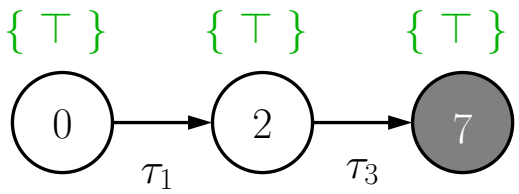
$$\tau_2\colon x \geq 1 \wedge \left\{ \begin{array}{l} x' := x - 1 \\ y' := y - 1 \end{array} \right.$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \left\{ \begin{array}{l} x' := x \\ y' := y \end{array} \right.$$

Lazy Abstraction with Interpolants

[McM06]

(Abstract) Program Unwinding



$\{ \top \}$

true

$x_1 = x_0$

$y_1 = x_0$

$\{ y_1 - x_1 \leq 0 \}$

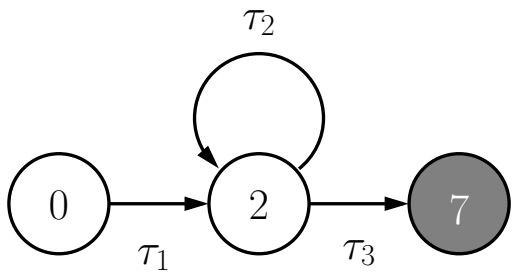
$x_1 \leq 0$

$y_1 \geq 1$

$x_2 = x_1$

$y_2 = y_1$

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \begin{cases} x' := x \\ y' := x \end{cases}$$

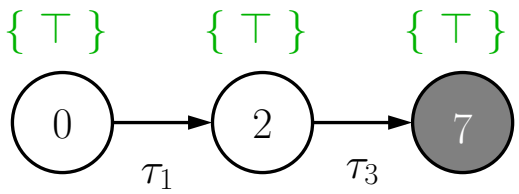
$$\tau_2\colon x \geq 1 \wedge \begin{cases} x' := x - 1 \\ y' := y - 1 \end{cases}$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \begin{cases} x' := x \\ y' := y \end{cases}$$

Lazy Abstraction with Interpolants

[McM06]

(Abstract) Program Unwinding



$\{ \top \}$

true

$x_1 = x_0$

$y_1 = x_0$

$\{ y_1 - x_1 \leq 0 \}$

$x_1 \leq 0$

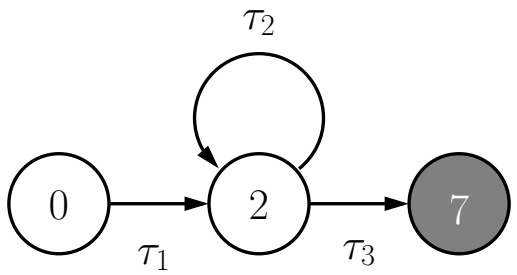
$y_1 \geq 1$

$x_2 = x_1$

$y_2 = y_1$

$\{ \bot \}$

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \begin{cases} x' := x \\ y' := x \end{cases}$$

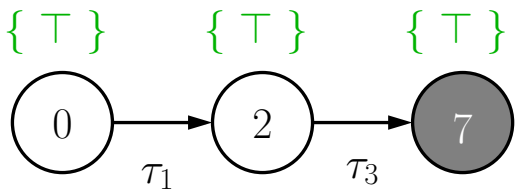
$$\tau_2\colon x \geq 1 \wedge \begin{cases} x' := x - 1 \\ y' := y - 1 \end{cases}$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \begin{cases} x' := x \\ y' := y \end{cases}$$

Lazy Abstraction with Interpolants

[McM06]

(Abstract) Program Unwinding



$A \Rightarrow I$

$I \wedge B$ is unsat

common vocabulary

{ ⊤ }

true

$x_1 = x_0$

$y_1 = x_0$

{ y_1 - $x_1 \leq 0$ }

$x_1 \leq 0$

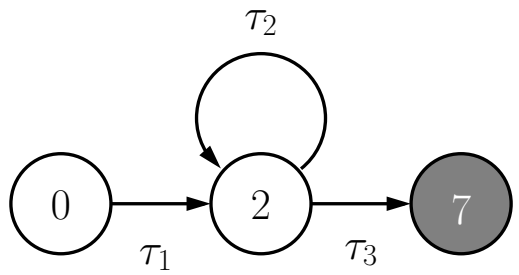
$y_1 \geq 1$

$x_2 = x_1$

$y_2 = y_1$

{ ⊥ }

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \begin{cases} x' := x \\ y' := x \end{cases}$$

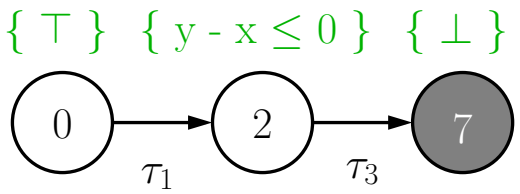
$$\tau_2\colon x \geq 1 \wedge \begin{cases} x' := x - 1 \\ y' := y - 1 \end{cases}$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \begin{cases} x' := x \\ y' := y \end{cases}$$

Lazy Abstraction with Interpolants

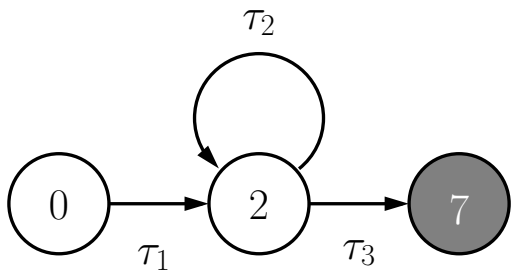
[McM06]

(Abstract) Program Unwinding



$\{ \top \}$
$\quad \text{true}$
$\quad x_1 = x_0$
$\quad y_1 = x_0$
$\{ y_1 - x_1 \leq 0 \}$
$\quad x_1 \leq 0$
$\quad y_1 \geq 1$
$\quad x_2 = x_1$
$\quad y_2 = y_1$
$\{ \bot \}$

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \begin{cases} x' := x \\ y' := x \end{cases}$$

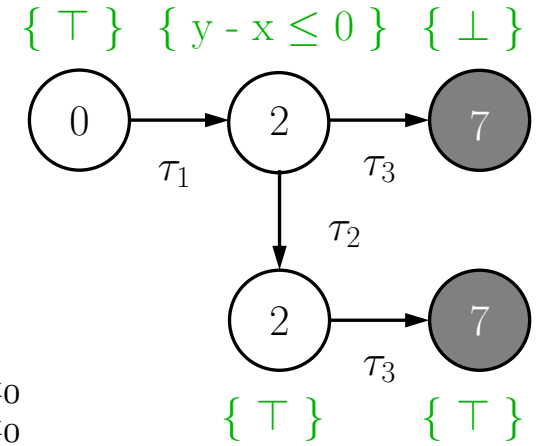
$$\tau_2\colon x \geq 1 \wedge \begin{cases} x' := x - 1 \\ y' := y - 1 \end{cases}$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \begin{cases} x' := x \\ y' := y \end{cases}$$

Lazy Abstraction with Interpolants

[McM06]

(Abstract) Program Unwinding

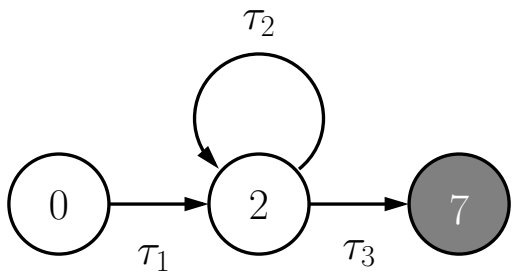


true
$x_1 = x_0$
$y_1 = x_0$

$x_1 \geq 1$
$x_2 = x_1 - 1$
$y_2 = y_1 - 1$

$x_2 \leq 0$
$y_2 \geq 1$
$x_3 = x_2$
$y_3 = y_2$

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \left\{ \begin{array}{l} x' := x \\ y' := x \end{array} \right.$$

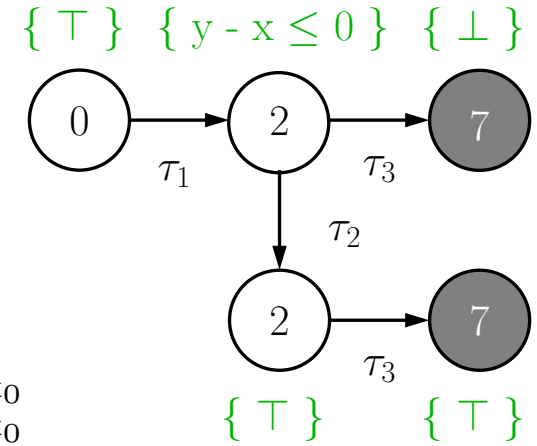
$$\tau_2\colon x \geq 1 \wedge \left\{ \begin{array}{l} x' := x - 1 \\ y' := y - 1 \end{array} \right.$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \left\{ \begin{array}{l} x' := x \\ y' := y \end{array} \right.$$

Lazy Abstraction with Interpolants

[McM06]

(Abstract) Program Unwinding

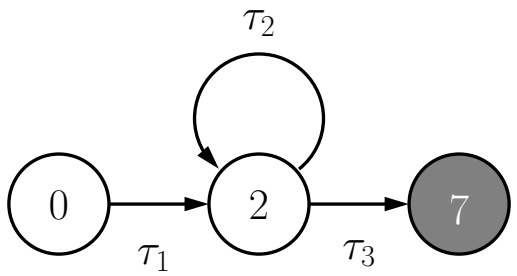


{ ⊤ }
true
$x_1 = x_0$
$y_1 = x_0$

$x_1 \geq 1$
$x_2 = x_1 - 1$
$y_2 = y_1 - 1$

$x_2 \leq 0$
$y_2 \geq 1$
$x_3 = x_2$
$y_3 = y_2$

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \left\{ \begin{array}{l} x' := x \\ y' := x \end{array} \right.$$

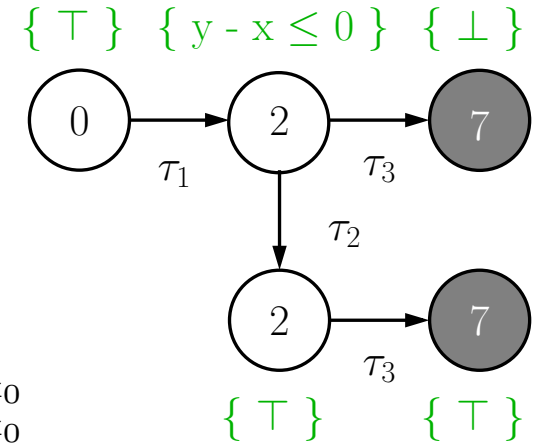
$$\tau_2\colon x \geq 1 \wedge \left\{ \begin{array}{l} x' := x - 1 \\ y' := y - 1 \end{array} \right.$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \left\{ \begin{array}{l} x' := x \\ y' := y \end{array} \right.$$

Lazy Abstraction with Interpolants

[McM06]

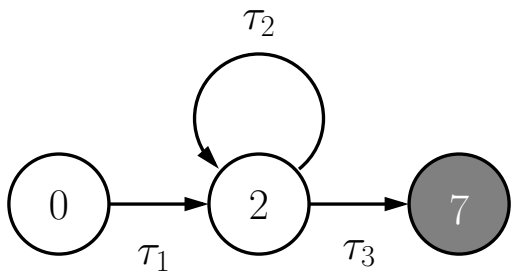
(Abstract) Program Unwinding



$\{ \top \}$
true
$x_1 = x_0$
$y_1 = x_0$
$\{ y_1 - x_1 \leq 0 \}$
$x_1 \geq 1$
$x_2 = x_1 - 1$
$y_2 = y_1 - 1$

$x_2 \leq 0$
$y_2 \geq 1$
$x_3 = x_2$
$y_3 = y_2$

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \left\{ \begin{array}{l} x' := x \\ y' := x \end{array} \right.$$

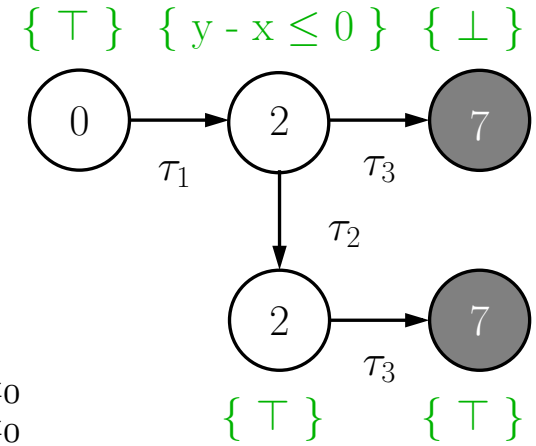
$$\tau_2\colon x \geq 1 \wedge \left\{ \begin{array}{l} x' := x - 1 \\ y' := y - 1 \end{array} \right.$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \left\{ \begin{array}{l} x' := x \\ y' := y \end{array} \right.$$

Lazy Abstraction with Interpolants

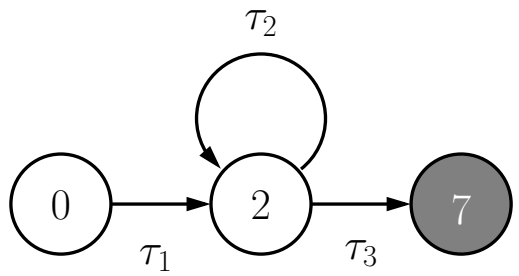
[McM06]

(Abstract) Program Unwinding



{ ⊤ }
true
$x_1 = x_0$
$y_1 = x_0$
{ $y_1 - x_1 \leq 0$ }
$x_1 \geq 1$
$x_2 = x_1 - 1$
$y_2 = y_1 - 1$
{ $y_2 - x_2 \leq 0$ }
$x_2 \leq 0$
$y_2 \geq 1$
$x_3 = x_2$
$y_3 = y_2$

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \left\{ \begin{array}{l} x' := x \\ y' := x \end{array} \right.$$

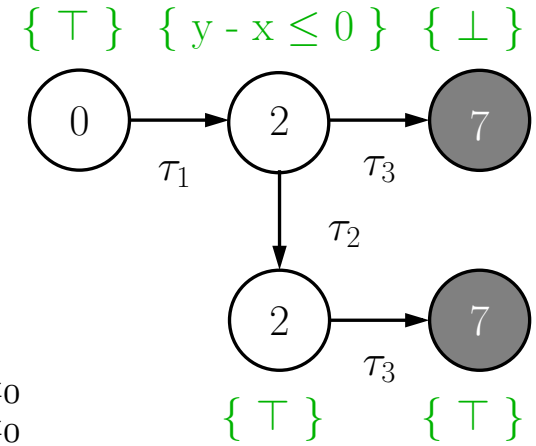
$$\tau_2\colon x \geq 1 \wedge \left\{ \begin{array}{l} x' := x - 1 \\ y' := y - 1 \end{array} \right.$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \left\{ \begin{array}{l} x' := x \\ y' := y \end{array} \right.$$

Lazy Abstraction with Interpolants

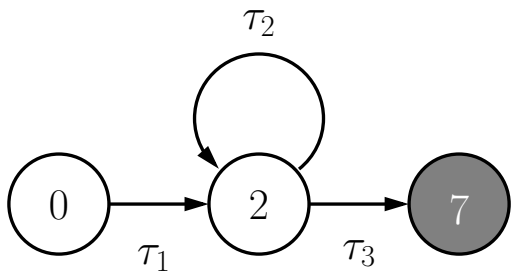
[McM06]

(Abstract) Program Unwinding



{ ⊤ }
  true
  $x_1 = x_0$
  $y_1 = x_0$
{ $y_1 - x_1 \leq 0$ }
  $x_1 \geq 1$
  $x_2 = x_1 - 1$
  $y_2 = y_1 - 1$
{ $y_2 - x_2 \leq 0$ }
  $x_2 \leq 0$
  $y_2 \geq 1$
  $x_3 = x_2$
  $y_3 = y_2$
{ ⊥ }

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \left\{ \begin{array}{l} x' := x \\ y' := x \end{array} \right.$$

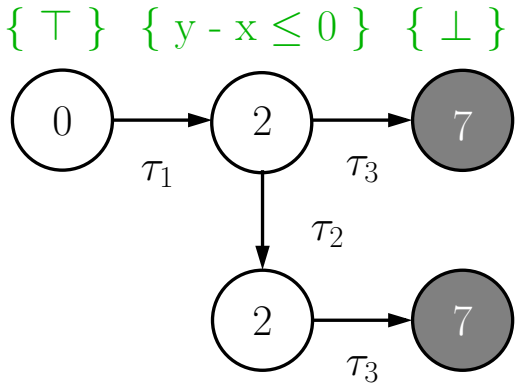
$$\tau_2\colon x \geq 1 \wedge \left\{ \begin{array}{l} x' := x - 1 \\ y' := y - 1 \end{array} \right.$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \left\{ \begin{array}{l} x' := x \\ y' := y \end{array} \right.$$

Lazy Abstraction with Interpolants

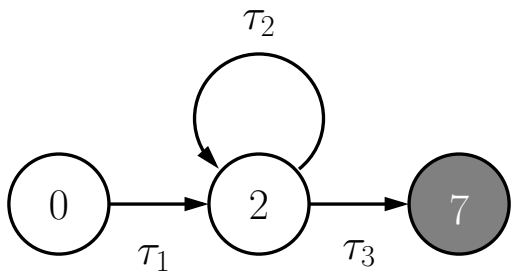
[McM06]

(Abstract) Program Unwinding



$\{ \top \}$
true
$x_1 = x_0$
$y_1 = x_0$
$\{ y_1 - x_1 \leq 0 \}$
$x_1 \geq 1$
$x_2 = x_1 - 1$
$y_2 = y_1 - 1$
$\{ y_2 - x_2 \leq 0 \}$
$x_2 \leq 0$
$y_2 \geq 1$
$x_3 = x_2$
$y_3 = y_2$
$\{ \bot \}$

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \left\{ \begin{array}{l} x' := x \\ y' := x \end{array} \right.$$

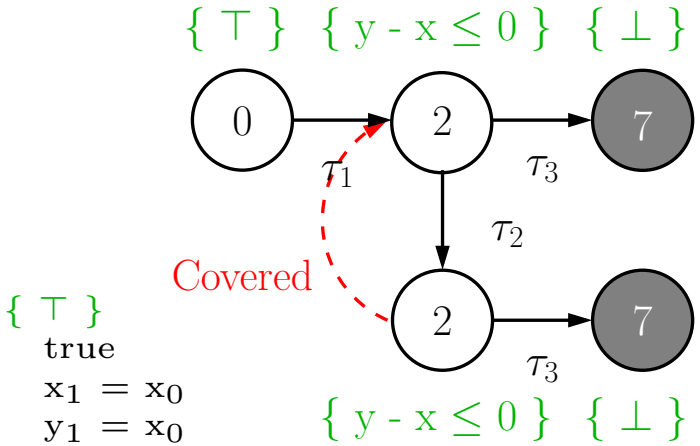
$$\tau_2\colon x \geq 1 \wedge \left\{ \begin{array}{l} x' := x - 1 \\ y' := y - 1 \end{array} \right.$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \left\{ \begin{array}{l} x' := x \\ y' := y \end{array} \right.$$

Lazy Abstraction with Interpolants

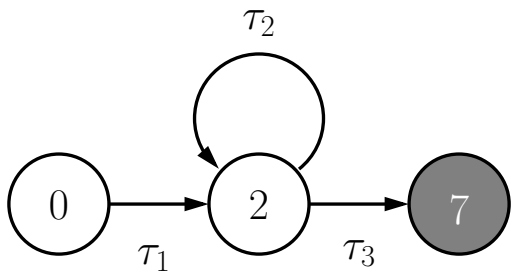
[McM06]

(Abstract) Program Unwinding



$\{ \top \}$
true
$x_1 = x_0$
$y_1 = x_0$
$\{ y_1 - x_1 \leq 0 \}$
$x_1 \geq 1$
$x_2 = x_1 - 1$
$y_2 = y_1 - 1$
$\{ y_2 - x_2 \leq 0 \}$
$x_2 \leq 0$
$y_2 \geq 1$
$x_3 = x_2$
$y_3 = y_2$
$\{ \bot \}$

Transition System (over $\mathcal{LIA}$)



$$\tau_1\colon \top \wedge \left\{ \begin{array}{l} x' := x \\ y' := x \end{array} \right.$$

$$\tau_2\colon x \geq 1 \wedge \left\{ \begin{array}{l} x' := x - 1 \\ y' := y - 1 \end{array} \right.$$

$$\tau_3\colon x \leq 0 \wedge y \geq 1 \wedge \left\{ \begin{array}{l} x' := x \\ y' := y \end{array} \right.$$

Lazy Abstraction with Interpolants for Arrays

[ABG+12a]

Extends the Lazy Abstraction with Interpolants framework [McM06]

Lazy Abstraction with Interpolants for Arrays

[ABG+12a]

Extends the Lazy Abstraction with Interpolants framework [McM06]

- allows array reasoning

Lazy Abstraction with Interpolants for Arrays

[ABG+12a]

Extends the Lazy Abstraction with Interpolants framework [McM06]

- allows array reasoning
- backward reachability analysis

Lazy Abstraction with Interpolants for Arrays

[ABG+12a]

Extends the Lazy Abstraction with Interpolants framework [McM06]

- allows array reasoning
- backward reachability analysis
- *(implicitly) quantified* formulas describing set of (backward) reachable states

Lazy Abstraction with Interpolants for Arrays

[ABG+12a]

Extends the Lazy Abstraction with Interpolants framework [McM06]

- allows array reasoning
- backward reachability analysis
- *(implicitly) quantified* formulas describing set of (backward) reachable states
- Refinement by means of *quantifier-free* interpolants (exploits existing interpolation algorithms)

Lazy Abstraction with Interpolants for Arrays

[ABG+12a]

Extends the Lazy Abstraction with Interpolants framework [McM06]

- allows array reasoning
- backward reachability analysis
- *(implicitly) quantified* formulas describing set of (backward) reachable states
- Refinement by means of *quantifier-free* interpolants (exploits existing interpolation algorithms)
- Implemented in SAFARI [ABG+12b]
  - ☞ Available on `www.verify.inf.usi.ch/safari`

Lazy Abstraction with Interpolants for Arrays
[ABG+12a]

Extends the Lazy Abstraction with Interpolants framework [McM06]

- allows array reasoning
- backward reachability analysis
- *(implicitly) quantified* formulas describing set of (backward) reachable states
- Refinement by means of *quantifier-free* interpolants (exploits existing interpolation algorithms)
- Implemented in SAFARI [ABG+12b]
  - ☞ Available on `www.verify.inf.usi.ch/safari`

Francesco Alberti, Roberto Bruttomesso, Silvio Ghilardi, Silvio Ranise, and Natasha Sharygina.

Lazy abstraction with interpolants for arrays.

In Nikolaj Bjørner and Andrei Voronkov, editors, *LPAR*, volume 7180 of *Lecture Notes in Computer Science*, pages 46–61. Springer, 2012.

Completeness (and Termination)

- Hypothesis ensuring the termination of SAFARI are rather restrictive

Completeness (and Termination)

- Hypothesis ensuring the termination of SAFARI are rather restrictive

- If they are not met, SAFARI is incomplete, but still sound
  - Only termination is involved

Completeness (and Termination)

- Hypothesis ensuring the termination of SAFARI are rather restrictive
- If they are not met, SAFARI is incomplete, but still sound
  - Only termination is involved
- What's the behavior of SAFARI in practice?

Heuristics

- ✘ Refinement suffer from several degrees of randomness

Heuristics

✘ Refinement suffer from several degrees of randomness

1. Counterexample Manipulation
   - Counterexamples may have multiple inconsistencies
   - Need to select "where" we want to refine

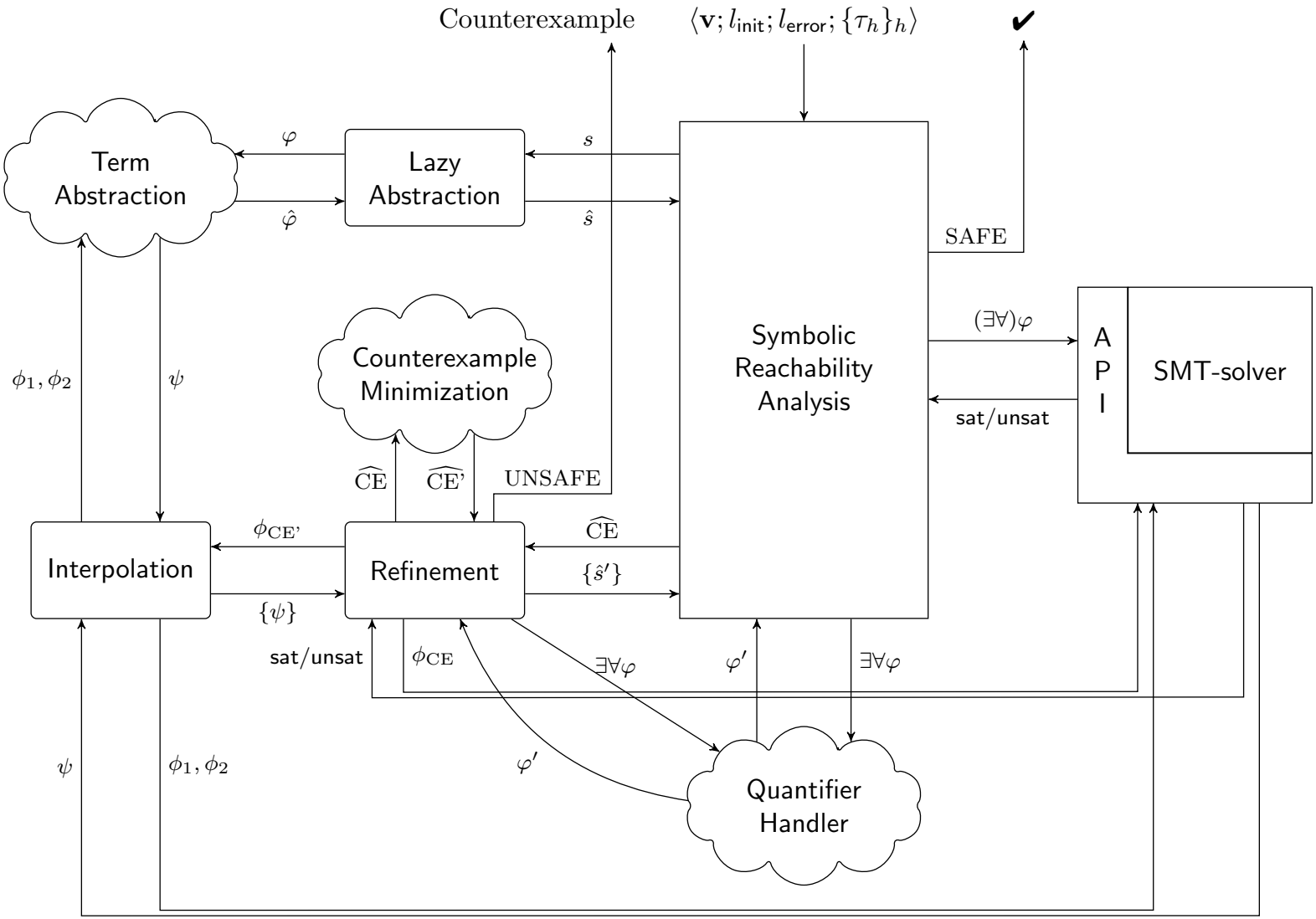
Heuristics

✘ Refinement suffer from several degrees of randomness

1. Counterexample Manipulation
   - Counterexamples may have multiple inconsistencies
   - Need to select "where" we want to refine

2. Interpolant Manipulation
   - Several interpolants for the inconsistent pair (A, B)
   - Need to select "how" we want to refine

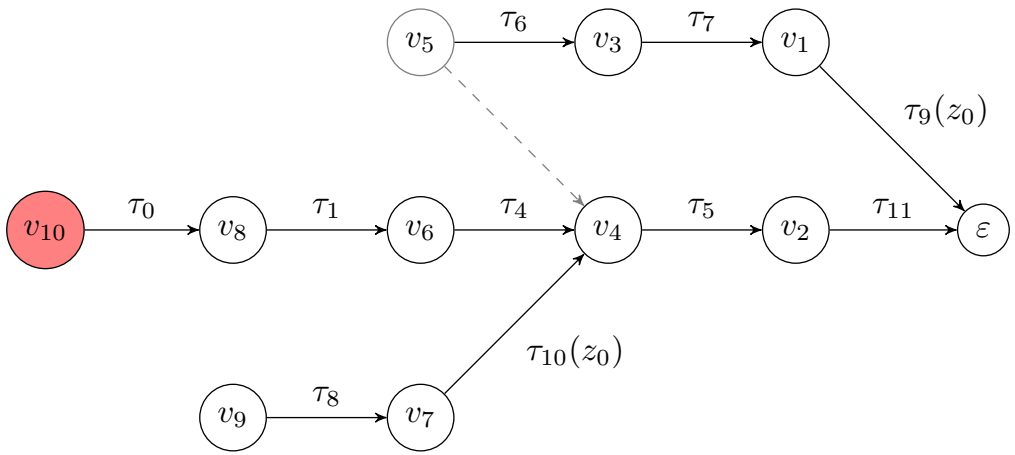
SAFARI architecture



Counterexample Manipulation

Counterexample minimization (CM)

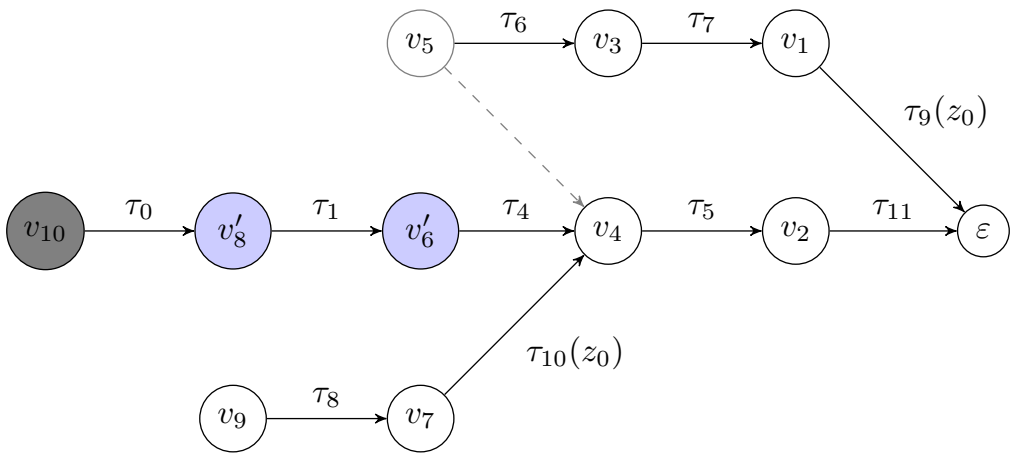
$$
\begin{aligned}
\tau_0 \quad & i_4 = 0 \wedge n > 0 \wedge pc_5 = 1 \wedge rv_4 = rv_5 \\
\tau_1 \quad & i_3 = i_4 \wedge 0 \leq i_4 \wedge rv_3 = rv_4 \wedge pc_4 = 2 \wedge \\
\tau_4 \quad & n \leq i_3 \wedge pc_3 = 3 \wedge i_2 = 0 \wedge rv_2 = \mathsf{true} \wedge \\
\tau_5 \quad & pc_2 = l_5 \wedge 0 \leq i_2 \wedge i_1 = i_2 \wedge rv_1 = rv_2 \wedge \\
\tau_{11} \quad & i_0 = i_1 \wedge n \leq i_1 \wedge rv_0 = \mathsf{false} \wedge rv_0 = rv_1 \wedge
\end{aligned}
$$



Counterexample Manipulation

Counterexample minimization (CM)

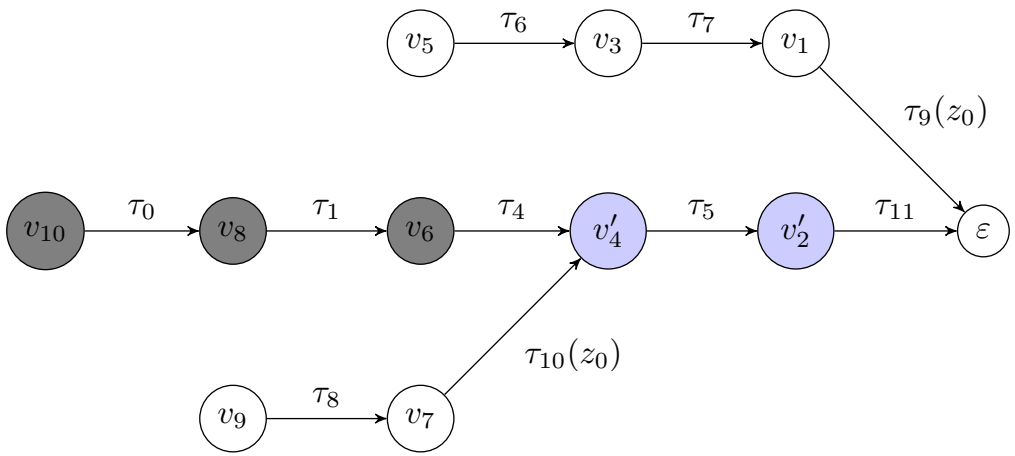
$$
\begin{aligned}
\tau_0 \quad & i_4 = 0 \wedge n > 0 \wedge pc_5 = 1 \wedge rv_4 = rv_5 \\
\tau_1 \quad & i_3 = i_4 \wedge 0 \leq i_4 \wedge rv_3 = rv_4 \wedge pc_4 = 2 \wedge \\
\tau_4 \quad & n \leq i_3 \wedge pc_3 = 3 \wedge i_2 = 0 \wedge rv_2 = \mathsf{true} \wedge \\
\tau_5 \quad & pc_2 = l_5 \wedge 0 \leq i_2 \wedge i_1 = i_2 \wedge rv_1 = rv_2 \wedge \\
\tau_{11} \quad & i_0 = i_1 \wedge n \leq i_1 \wedge rv_0 = \mathsf{false} \wedge rv_0 = rv_1 \wedge
\end{aligned}
$$



Counterexample Manipulation

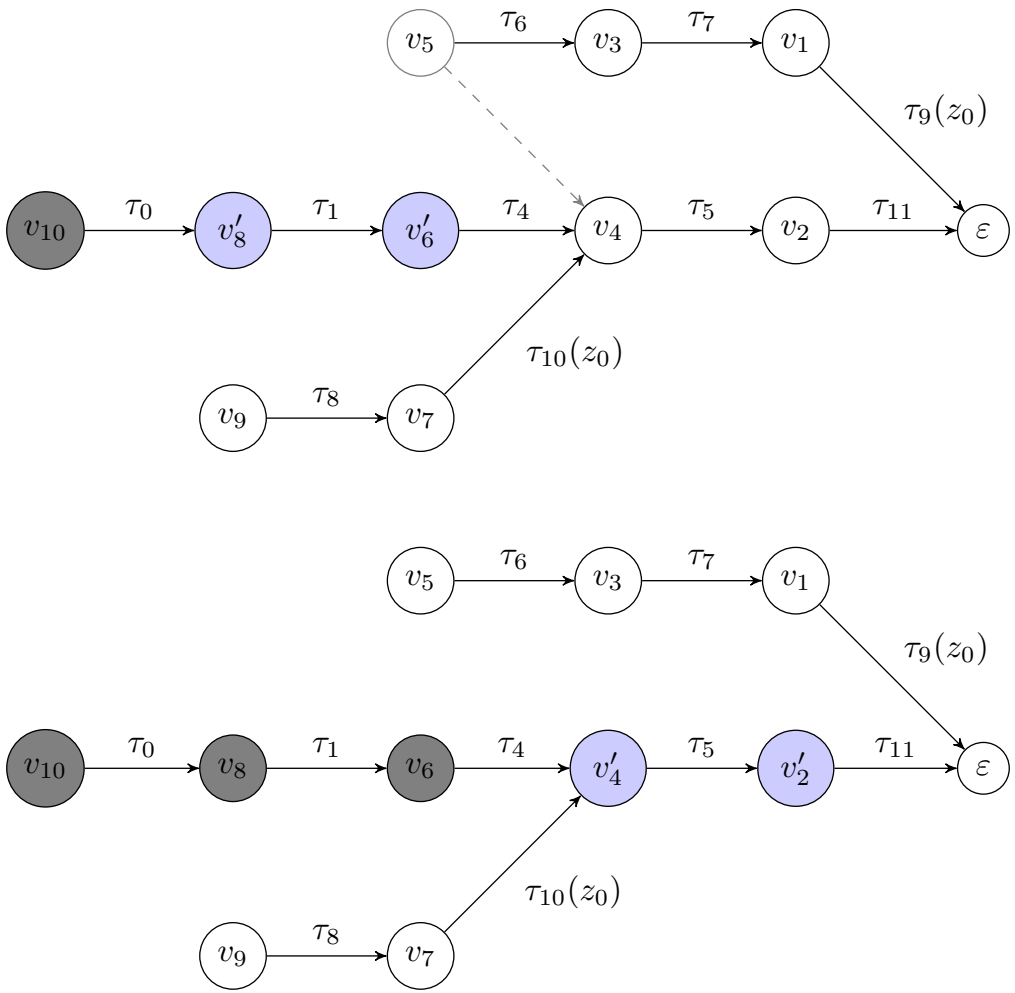
Counterexample minimization (CM)

$$
\begin{aligned}
\tau_0 \quad & i_4 = 0 \wedge n > 0 \wedge pc_5 = 1 \wedge rv_4 = rv_5 \\
\tau_1 \quad & i_3 = i_4 \wedge 0 \leq i_4 \wedge rv_3 = rv_4 \wedge pc_4 = 2 \wedge \\
\tau_4 \quad & n \leq i_3 \wedge pc_3 = 3 \wedge i_2 = 0 \wedge rv_2 = \mathsf{true} \wedge \\
\tau_5 \quad & pc_2 = l_5 \wedge 0 \leq i_2 \wedge i_1 = i_2 \wedge rv_1 = rv_2 \wedge \\
\tau_{11} \quad & i_0 = i_1 \wedge n \leq i_1 \wedge rv_0 = \mathsf{false} \wedge rv_0 = rv_1 \wedge
\end{aligned}
$$



Counterexample Manipulation

Counterexample minimization (CM)



Interpolant Manipulation

- Multiple possible interpolants

Interpolant Manipulation

- Multiple possible interpolants
- Different effect on verification performance, convergence

Interpolant Manipulation

- Multiple possible interpolants
- Different effect on verification performance, convergence
- Heuristics:

Interpolant Manipulation

- Multiple possible interpolants
- Different effect on verification performance, convergence
- Heuristics:
  - Interpolant strength variation

Interpolant Manipulation

- Multiple possible interpolants
- Different effect on verification performance, convergence
- Heuristics:
  - Interpolant strength variation
  - Theory interpolant manipulation

Interpolant Manipulation

- Multiple possible interpolants
- Different effect on verification performance, convergence
- Heuristics:
  - Interpolant strength variation
  - Theory interpolant manipulation
  - Term Abstraction

Interpolant strength variation

- I_1 stronger than I_2 $\qquad\qquad$ $I_1 \rightarrow I_2$

Interpolant strength variation

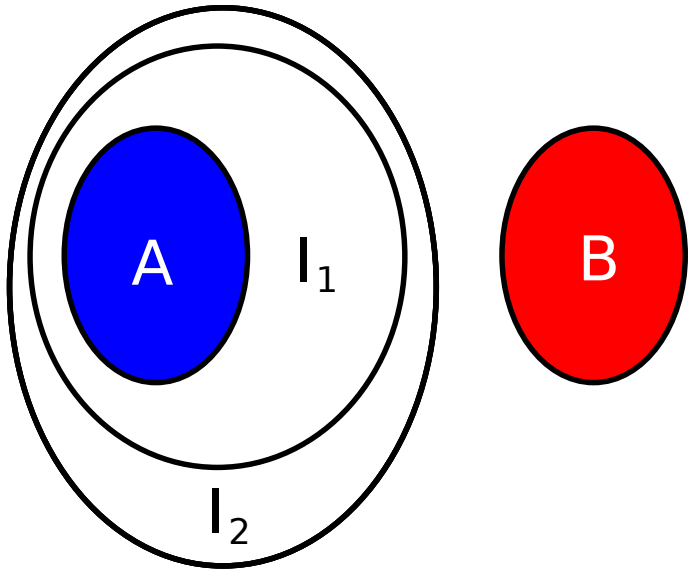
- I_1 stronger than I_2 $\qquad\qquad$ $I_1 \rightarrow I_2$
- Strength affects approximation coarseness

Interpolant strength variation

- I_1 stronger than I_2 $\qquad\qquad$ $I_1 \rightarrow I_2$
- Strength affects approximation coarseness
- Strength can affect convergence, performance

Interpolant strength variation

- I_1 stronger than I_2 $\qquad\qquad$ $I_1 \rightarrow I_2$
- Strength affects approximation coarseness
- Strength can affect convergence, performance



Labeled Interpolation Systems

Propositional Interpolation

- Interpolation parametric in labeling function [DKPW10]

Labeled Interpolation Systems

Propositional Interpolation

- Interpolation parametric in labeling function [DKPW10]
- Interpolant determined by proof and labeling L

Labeled Interpolation Systems

Propositional Interpolation

- Interpolation parametric in labeling function [DKPW10]
- Interpolant determined by proof and labeling L
- Generalization of [Pud97, McM04] (P, M, M')

Labeled Interpolation Systems

Propositional Interpolation

- Interpolation parametric in labeling function [DKPW10]
- Interpolant determined by proof and labeling L
- Generalization of [Pud97, McM04] (P, M, M')
- Strength comparison reduced to labeling comparison

Labeling Lattice

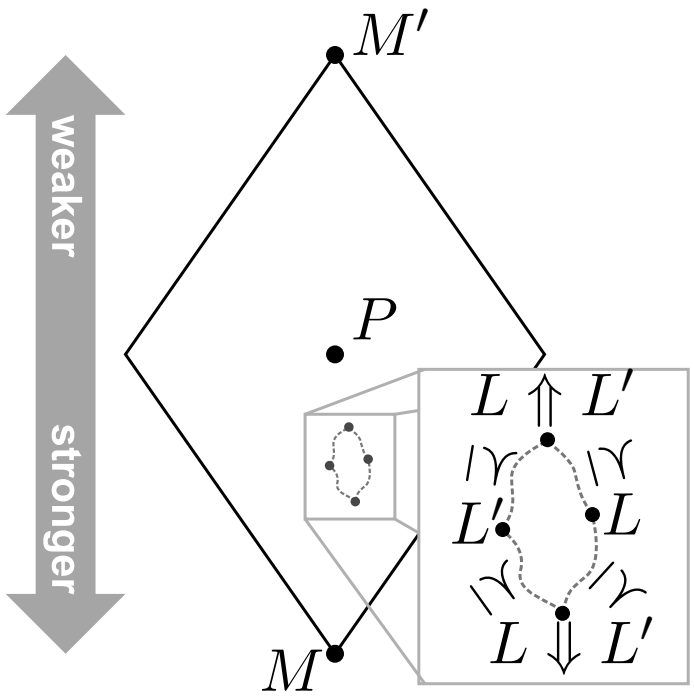
Labeled Interpolation Systems

- $L_1 \preceq L_2 \qquad \Longrightarrow \qquad I_1 \rightarrow I_2$

Labeling Lattice

Labeled Interpolation Systems

- $L_1 \preceq L_2 \quad \Longrightarrow \quad I_1 \rightarrow I_2$
- Labeling lattice



Path Interpolation

Labeled Interpolation Systems

- Spurious counterexample $\tau_1 \wedge \ldots \wedge \tau_n$

Path Interpolation

Labeled Interpolation Systems

- Spurious counterexample $\tau_1 \wedge \ldots \wedge \tau_n$
- Generation of multiple interpolants $I_1, \ldots, I_n$

Path Interpolation

Labeled Interpolation Systems

- Spurious counterexample $\tau_1 \wedge \ldots \wedge \tau_n$
- Generation of multiple interpolants $I_1, \ldots, I_n$
- Requirement $\tau_1 \rightarrow I_1 \qquad I_i \wedge \tau_{i+1} \rightarrow I_{i+1} \qquad I_{n-1} \wedge \tau_n \rightarrow \bot$

Path Interpolation

Labeled Interpolation Systems

- Spurious counterexample $\tau_1 \wedge \ldots \wedge \tau_n$
- Generation of multiple interpolants $I_1, \ldots, I_n$
- Requirement $\tau_1 \rightarrow I_1 \qquad I_i \wedge \tau_{i+1} \rightarrow I_{i+1} \qquad I_{n-1} \wedge \tau_n \rightarrow \bot$
- Generation of each I_i with different L_i

Path Interpolation

Labeled Interpolation Systems

- Spurious counterexample $\tau_1 \wedge \ldots \wedge \tau_n$
- Generation of multiple interpolants $I_1, \ldots, I_n$
- Requirement $\tau_1 \rightarrow I_1 \qquad I_i \wedge \tau_{i+1} \rightarrow I_{i+1} \qquad I_{n-1} \wedge \tau_n \rightarrow \bot$
- Generation of each I_i with different L_i
- Identification of constraints on $L_1, \ldots, L_n$

Path Interpolation

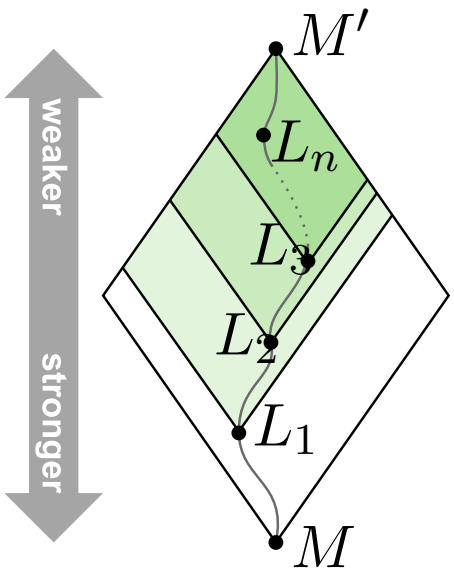
Propositional Constraints

- Requirement $\tau_1 \rightarrow I_1$ $I_i \wedge \tau_{i+1} \rightarrow I_{i+1}$ $I_{n-1} \wedge \tau_n \rightarrow \bot$

Path Interpolation

Propositional Constraints

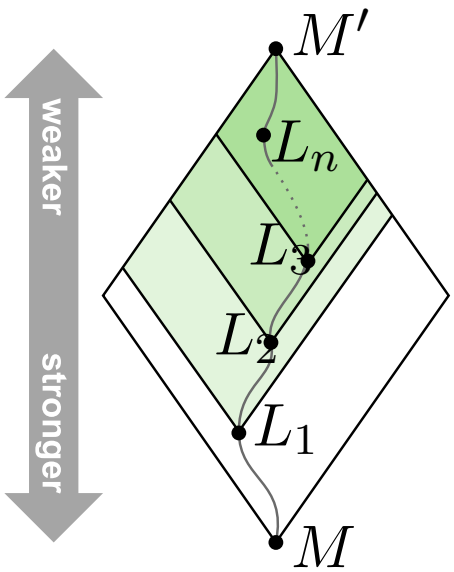
- Requirement $\tau_1 \to I_1$ $I_i \wedge \tau_{i+1} \to I_{i+1}$ $I_{n-1} \wedge \tau_n \to \bot$
- Satisfied for $L_1 \preceq \ldots \preceq L_n$ [RSS12]



Path Interpolation

Propositional Constraints

- Requirement $\tau_1 \to I_1 \qquad I_i \wedge \tau_{i+1} \to I_{i+1} \qquad I_{n-1} \wedge \tau_n \to \bot$
- Satisfied for $L_1 \preceq \ldots \preceq L_n$ [RSS12]
- Always satisfied for $L_1 \equiv \ldots \equiv L_n$ [GRS13]



- Interpolant I for unsatisfiable $A \wedge B$

- Interpolant I for unsatisfiable $A \wedge B$
- Generation [YM05]

- Interpolant I for unsatisfiable $A \wedge B$
- Generation [YM05]
  - Derivation of unsatisfiability resolution proof of $A \wedge B$

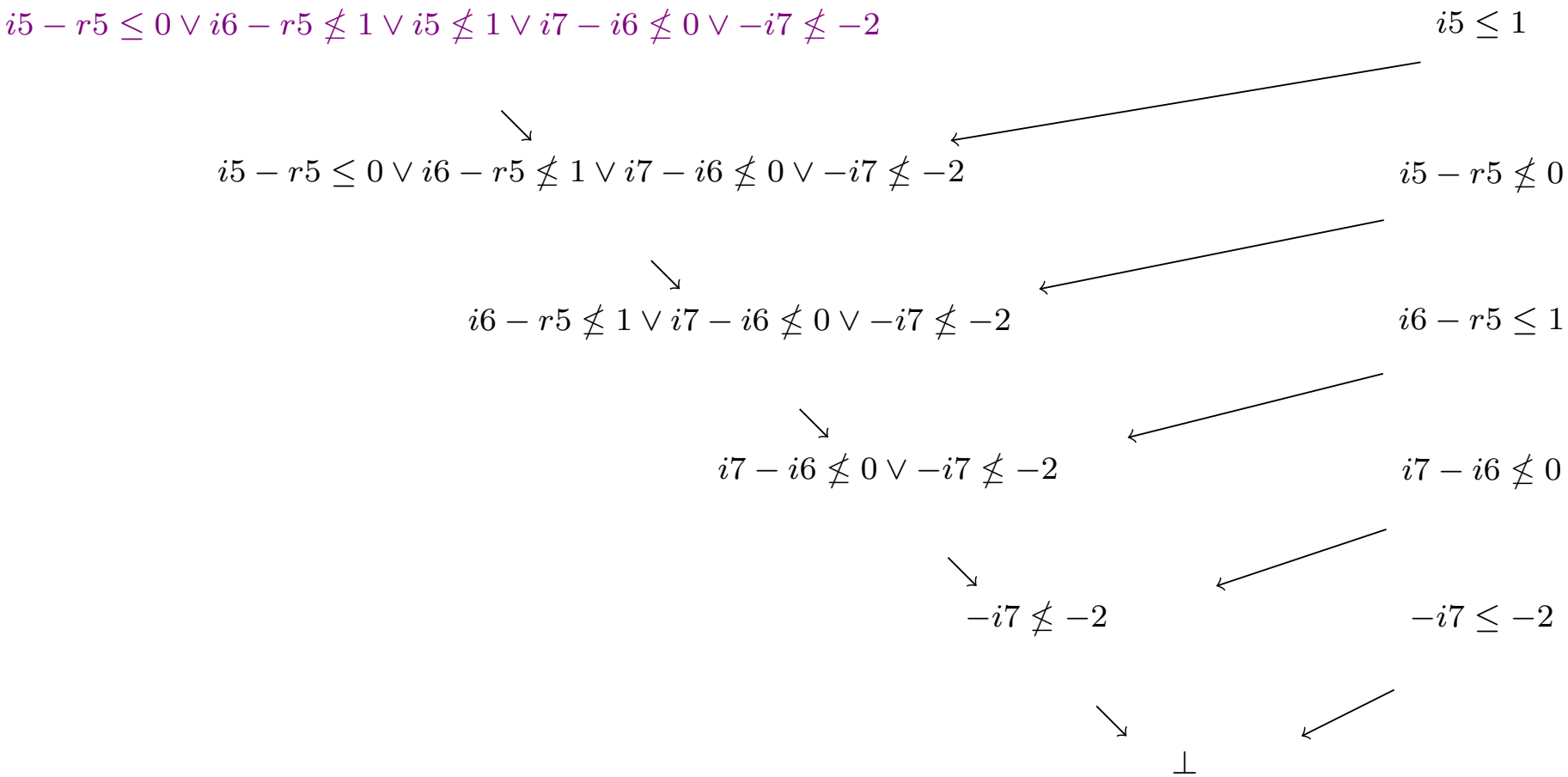
- Interpolant I for unsatisfiable $A \wedge B$
- Generation [YM05]
  - Derivation of unsatisfiability resolution proof of $A \wedge B$
  - Computation of I from proof structure

- Interpolant I for unsatisfiable $A \wedge B$
- Generation [YM05]
  - Derivation of unsatisfiability resolution proof of $A \wedge B$
  - Computation of I from proof structure
    - Partial interpolants for original clauses, theory interpolants for theory lemmata

- Interpolant I for unsatisfiable $A \wedge B$
- Generation [YM05]
  - Derivation of unsatisfiability resolution proof of $A \wedge B$
  - Computation of I from proof structure
    - Partial interpolants for original clauses, theory interpolants for theory lemmata
    - Partial interpolants for inner nodes

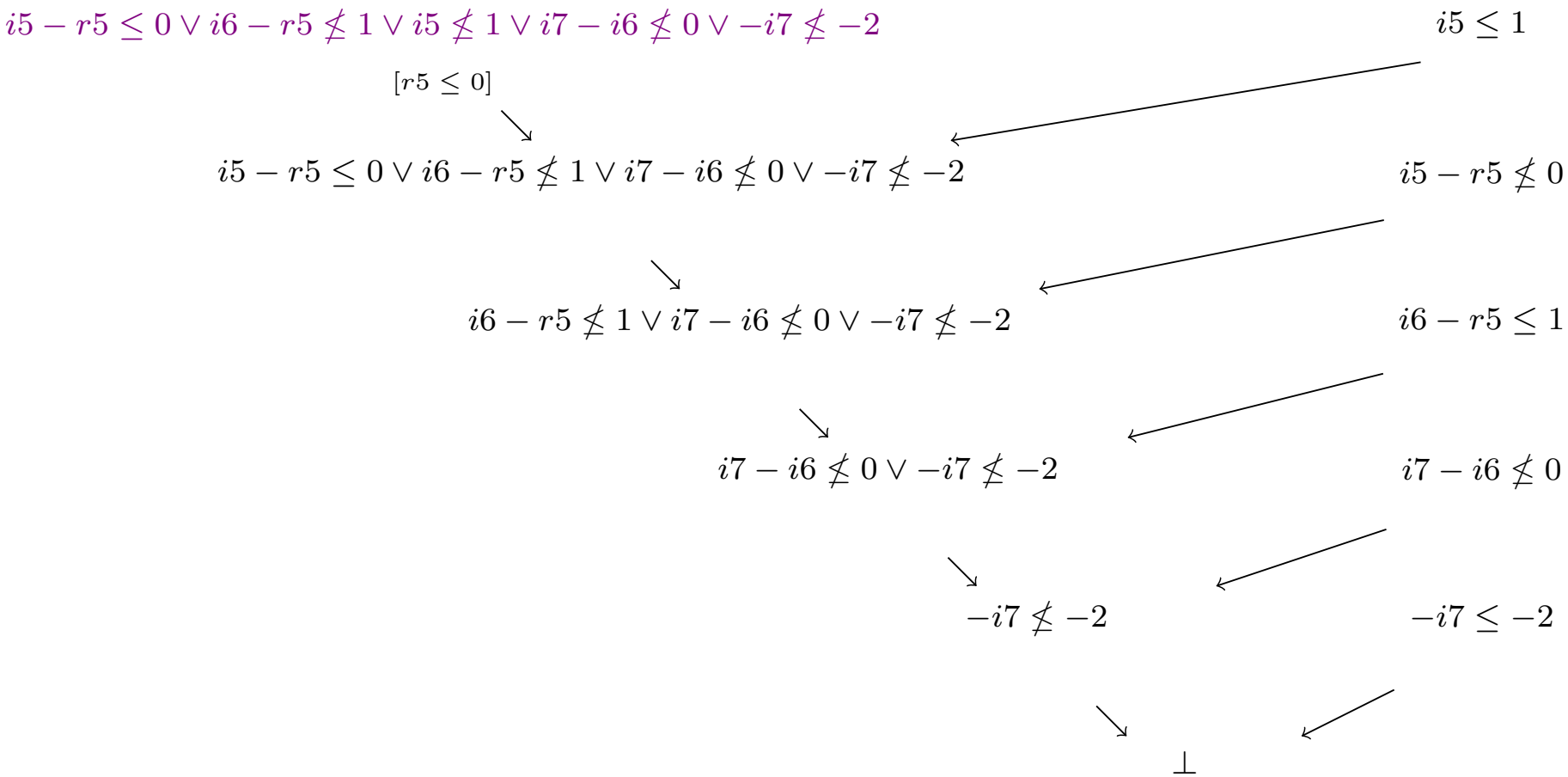
Interpolation in SMT

Example



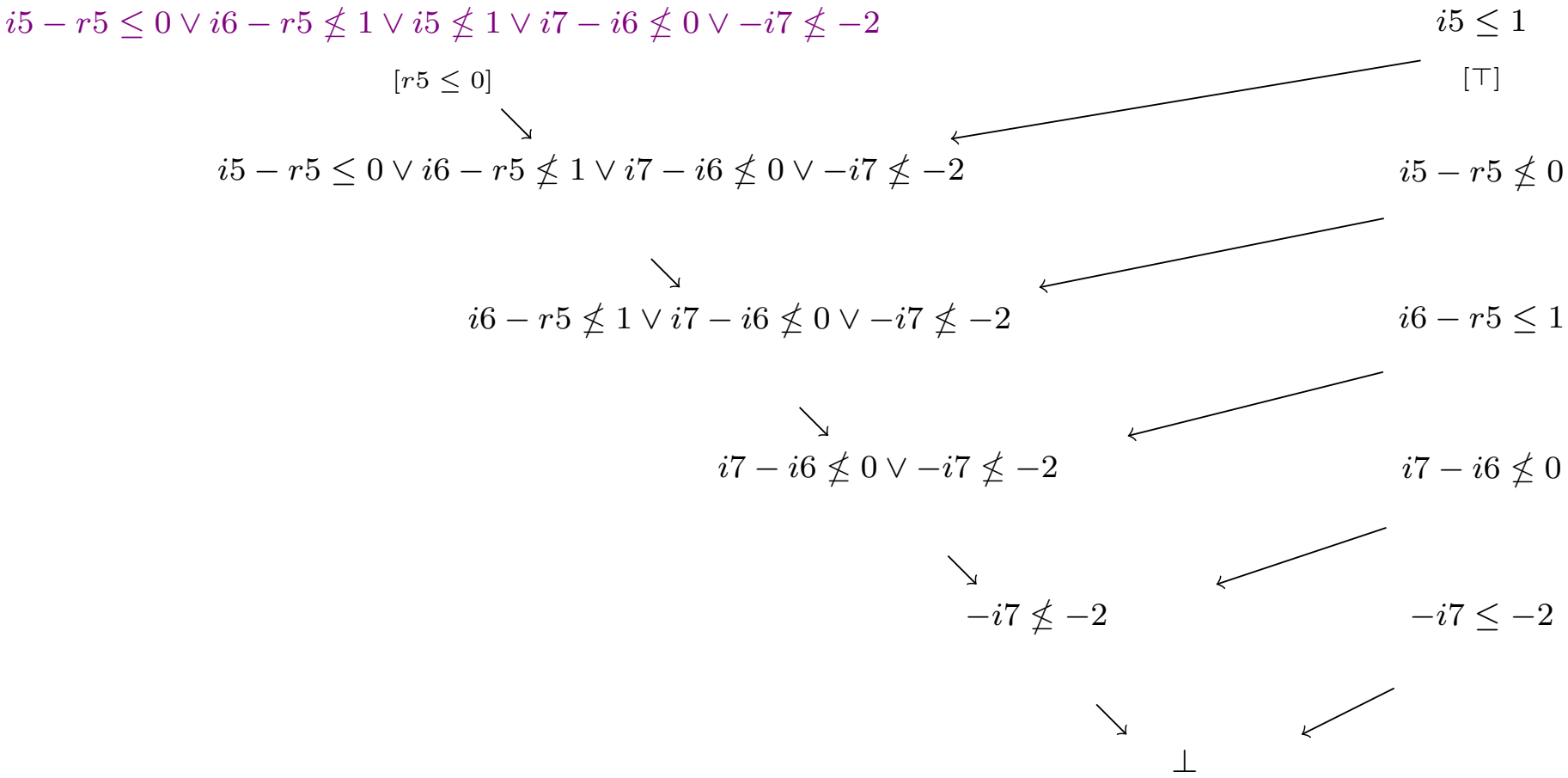
Interpolation in SMT

Example



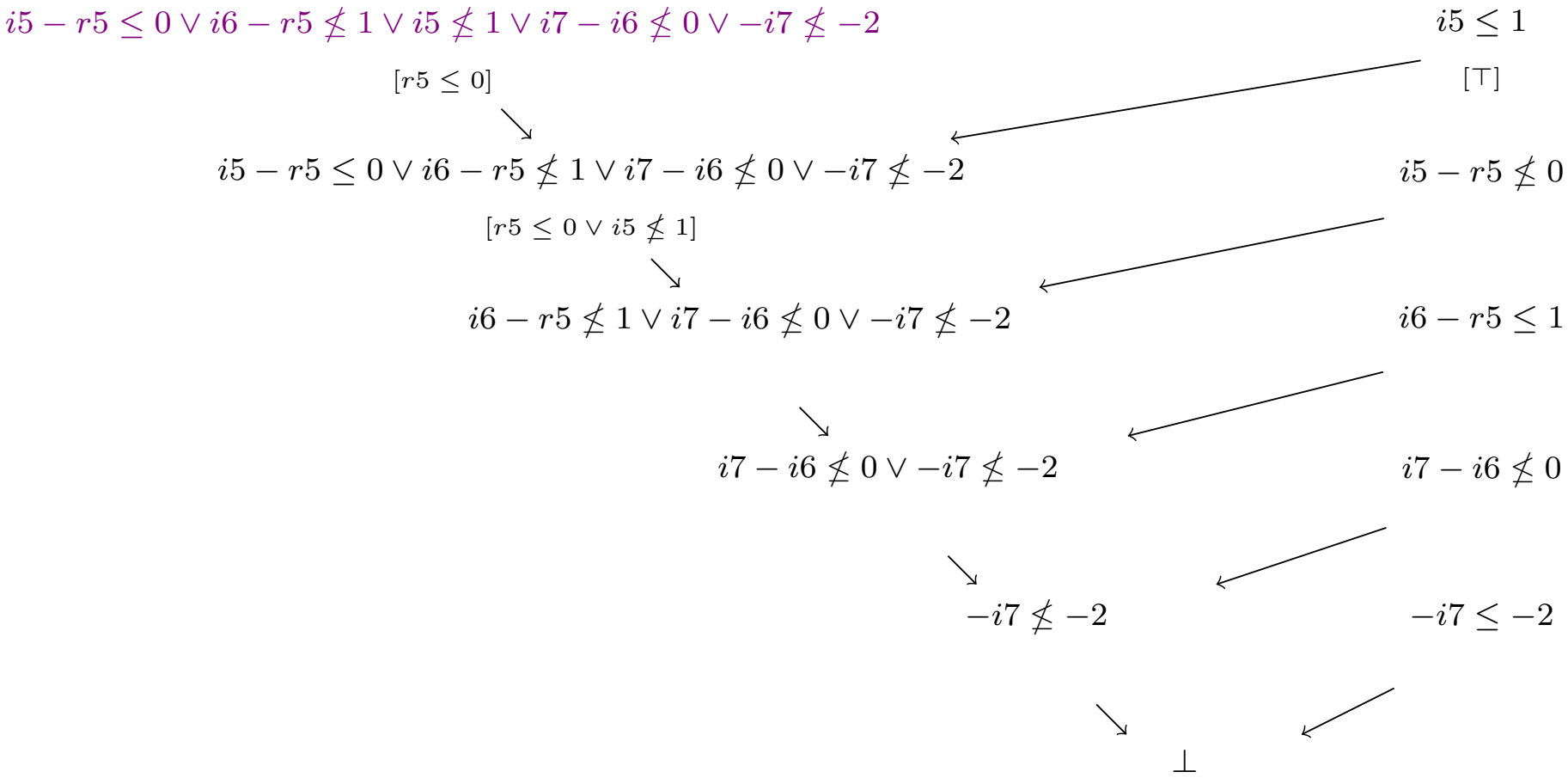
Interpolation in SMT

Example



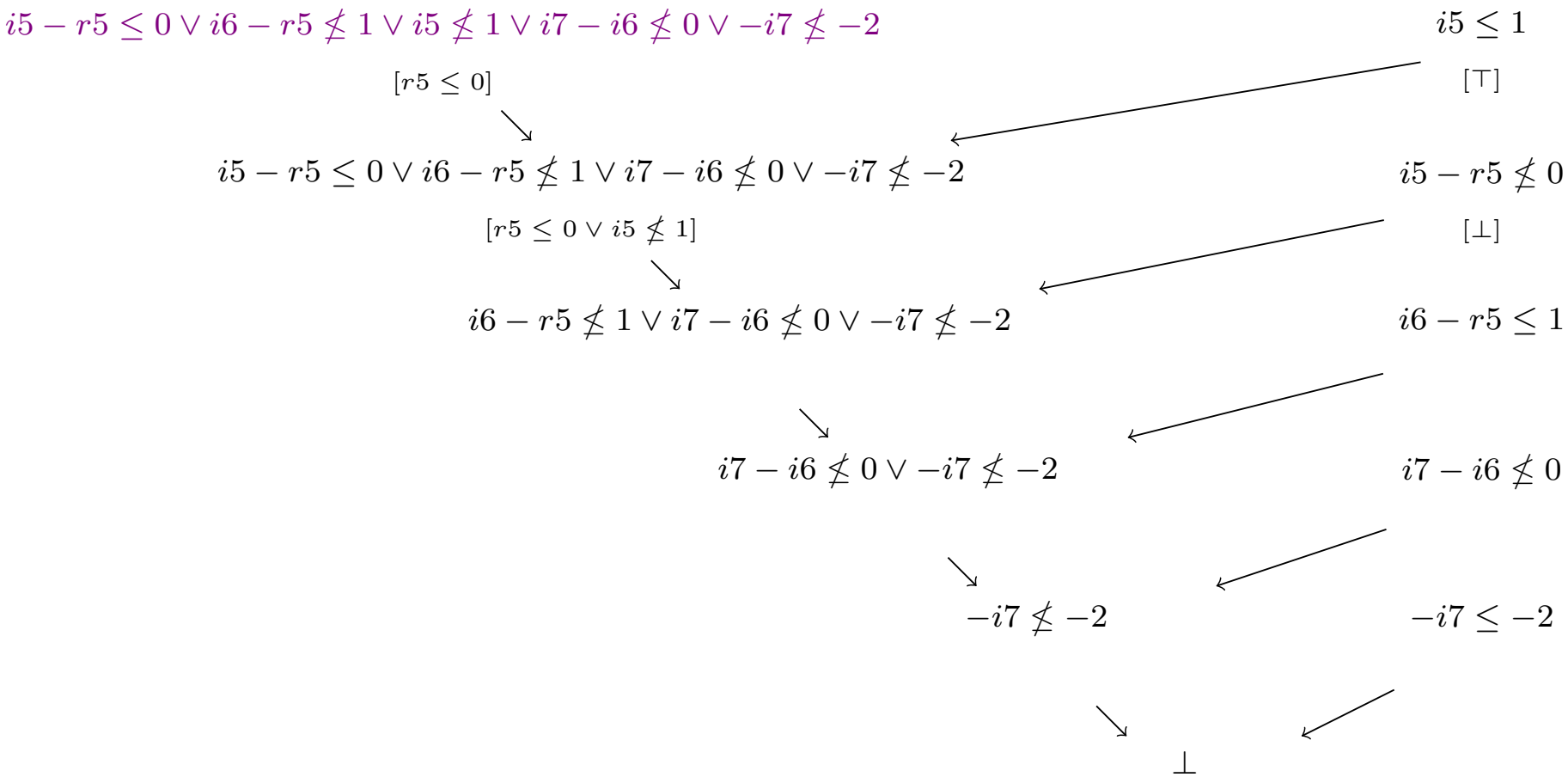
Interpolation in SMT

Example



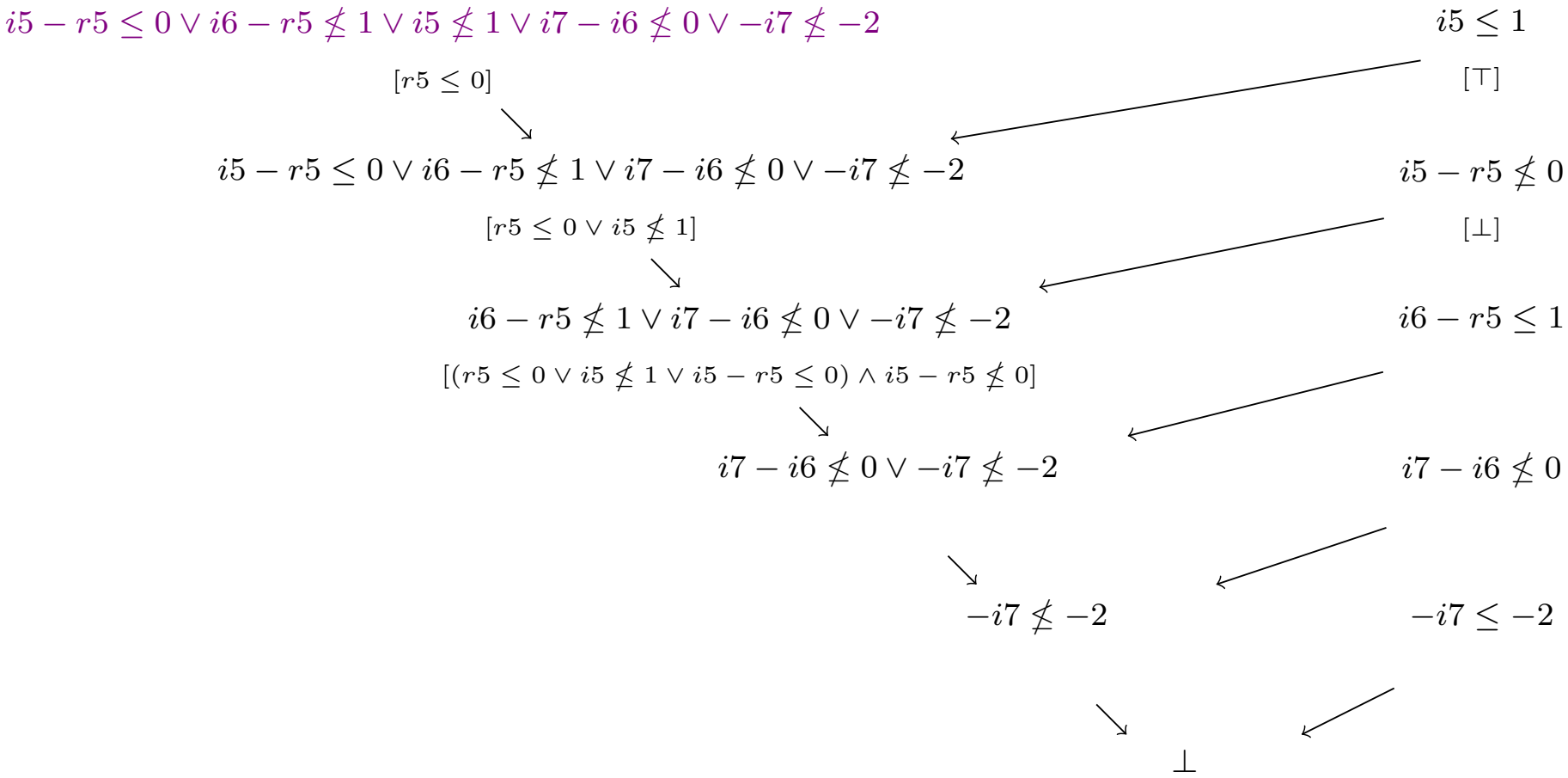
Interpolation in SMT

Example



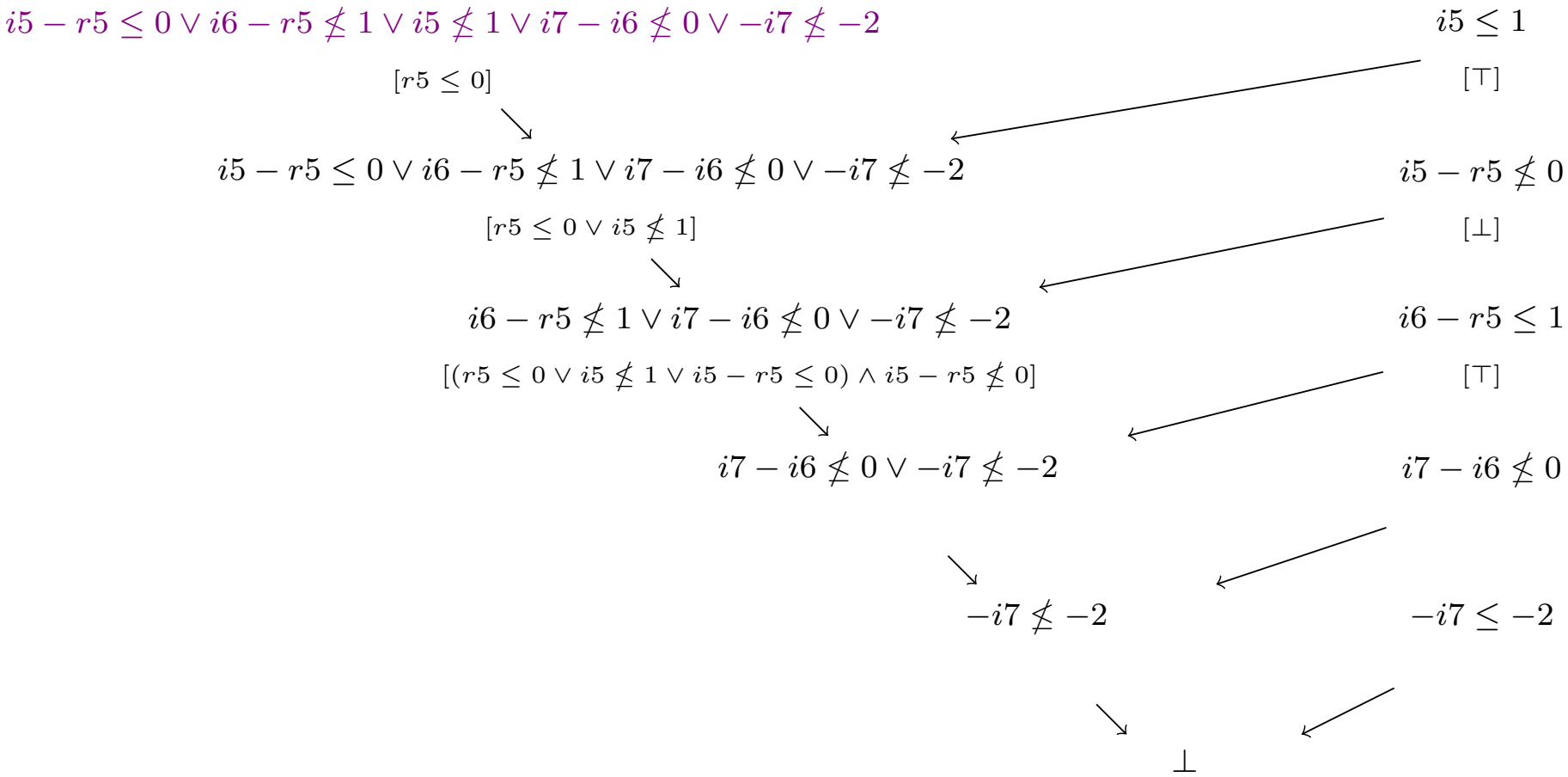
Interpolation in SMT

Example



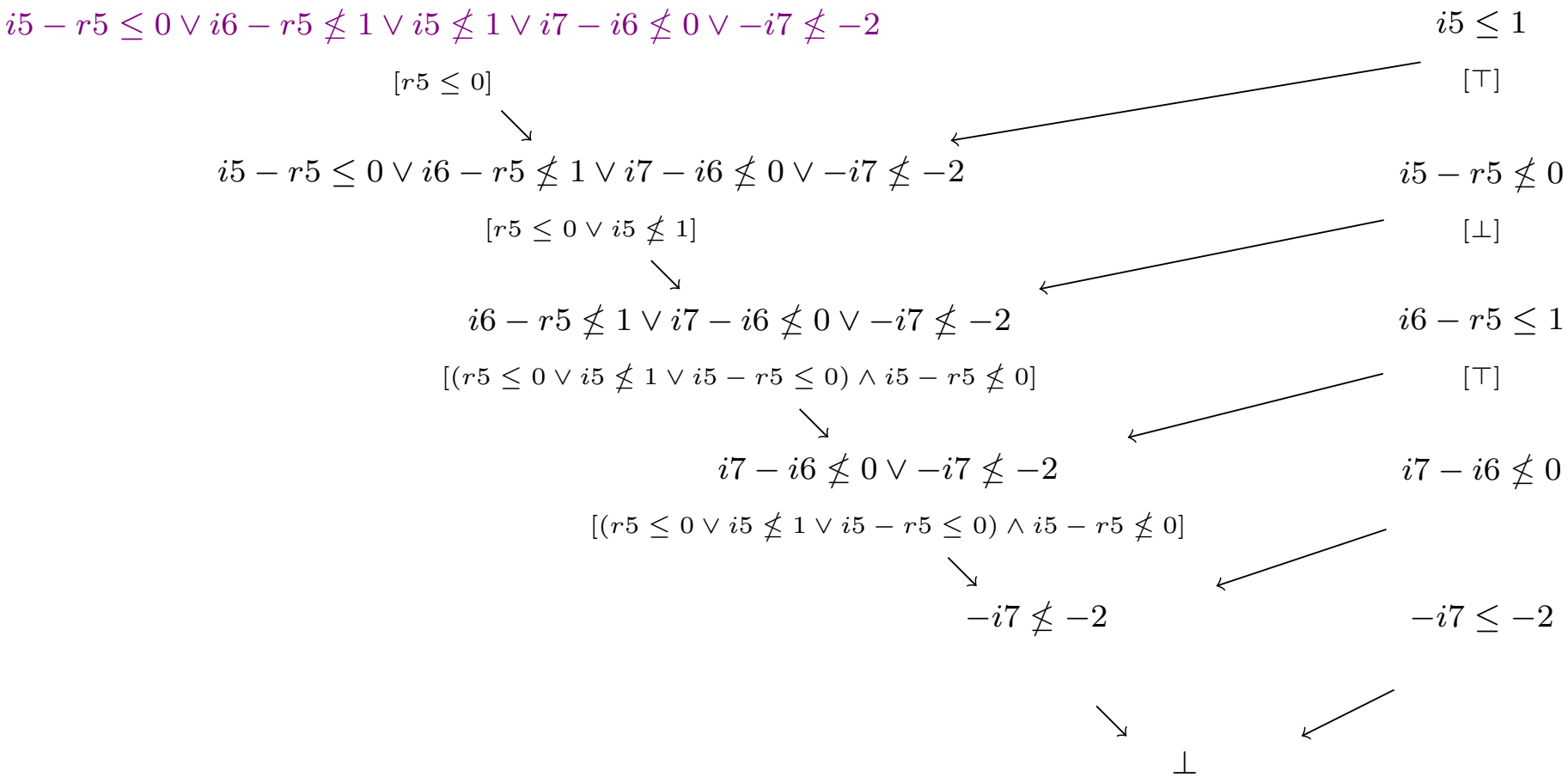
Interpolation in SMT

Example



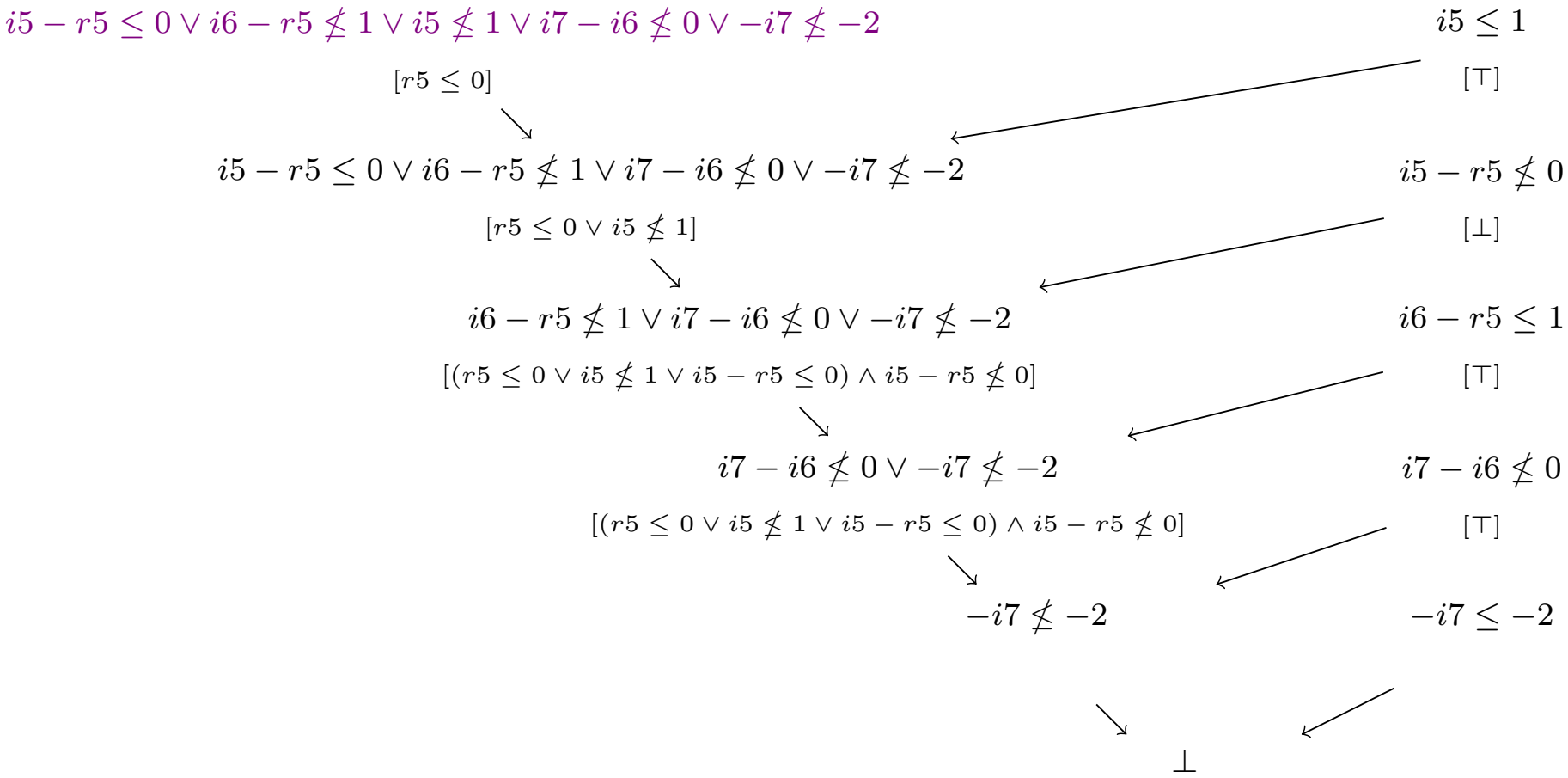
Interpolation in SMT

Example



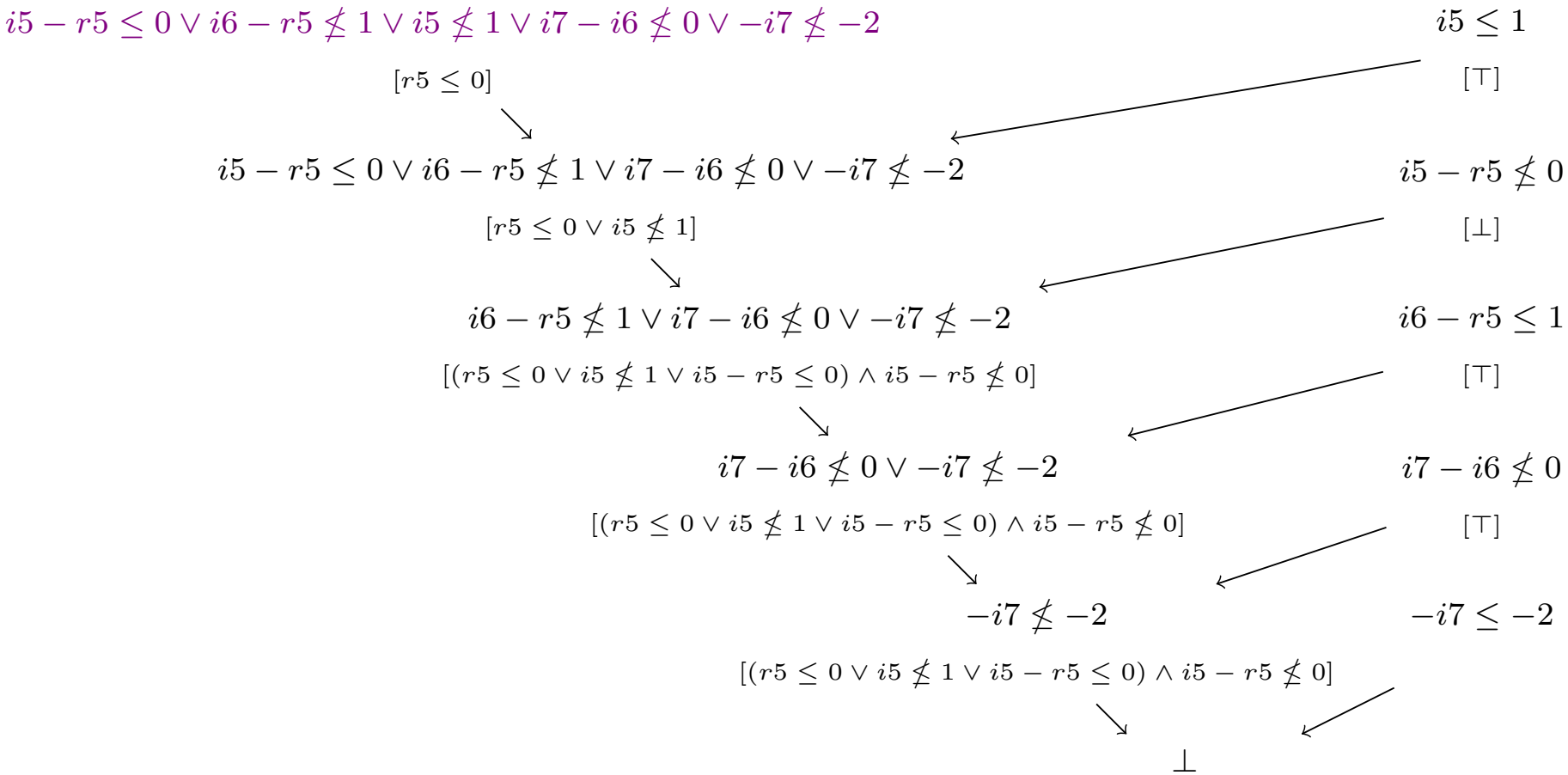
Interpolation in SMT

Example



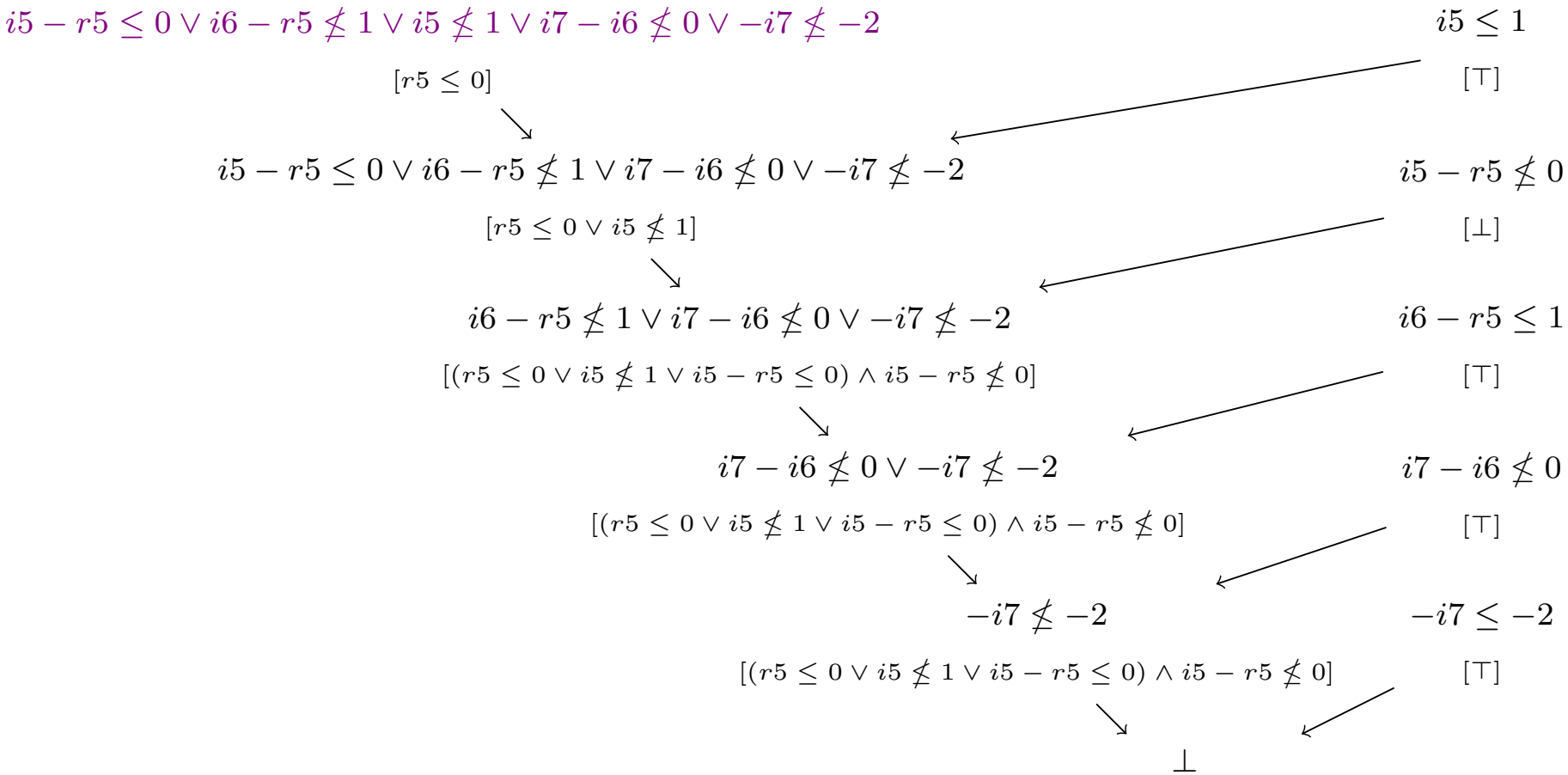
Interpolation in SMT

Example



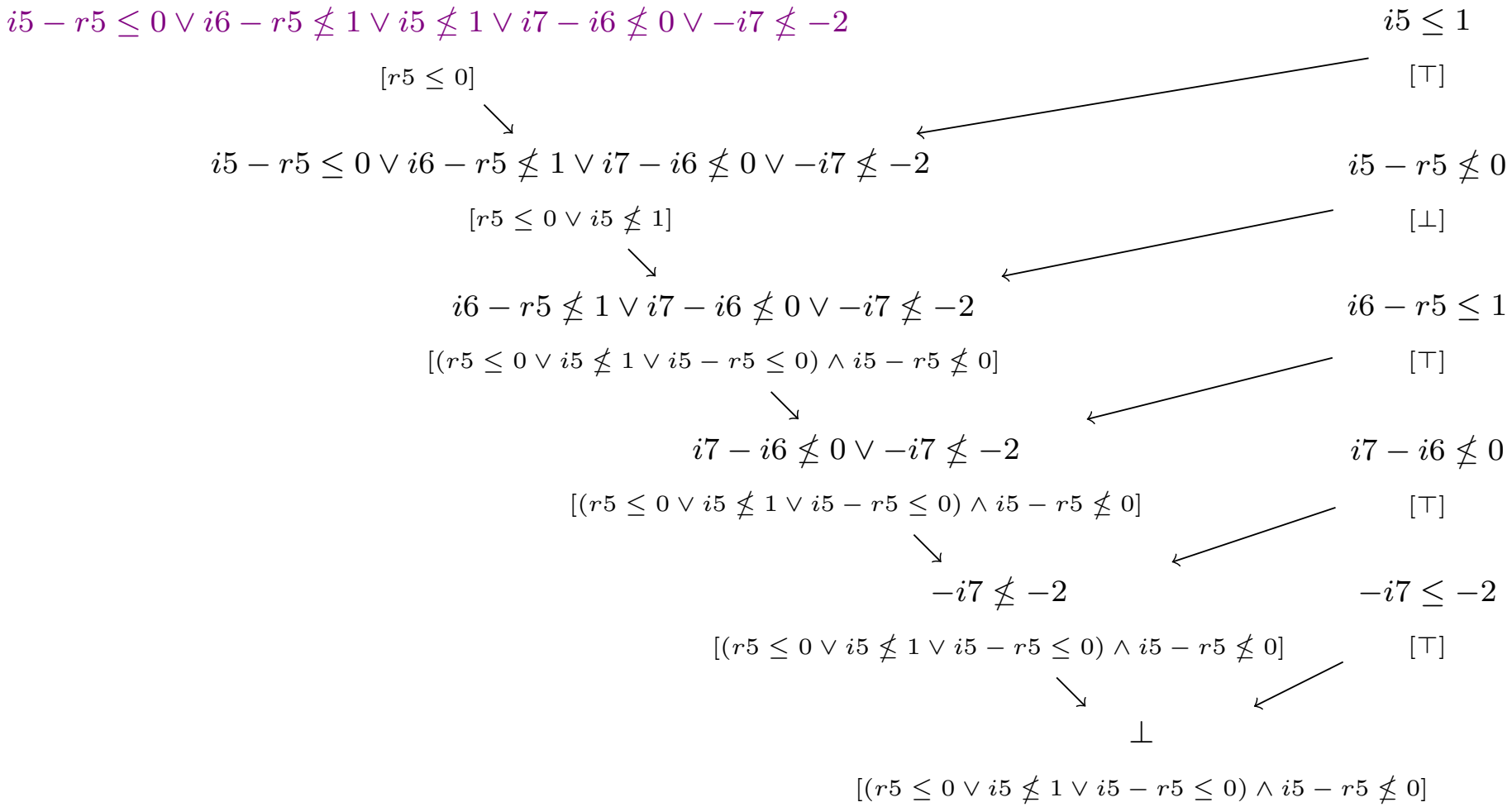
Interpolation in SMT

Example



Interpolation in SMT

Example



$\mathcal{I}$nteger $\mathcal{D}$ifference $\mathcal{L}$ogic:

- Atoms are inequalities of the form $x \leq y + c$

Interpolation in SMT

Theory interpolants in $\mathcal{IDL}$

$\mathcal{I}$nteger $\mathcal{D}$ifference $\mathcal{L}$ogic:

- Atoms are inequalities of the form $x \leq y + c$
- Theory lemma $\mapsto$ inconsistent set of theory atoms

Interpolation in SMT

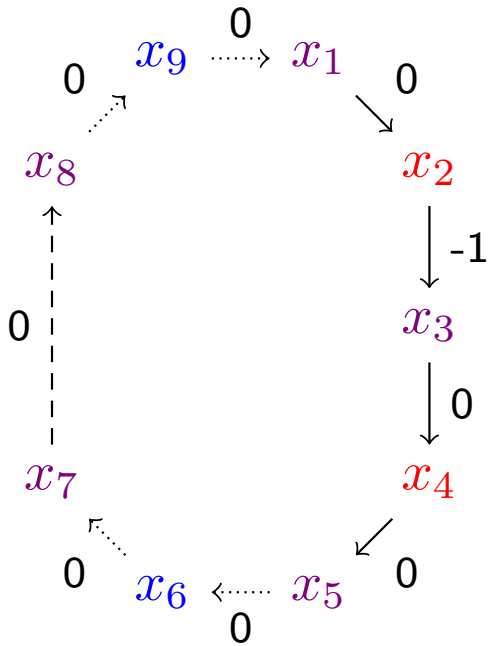
Theory interpolants in $\mathcal{IDL}$

$\mathcal{I}$nteger $\mathcal{D}$ifference $\mathcal{L}$ogic:

- Atoms are inequalities of the form $x \leq y + c$
- Theory lemma $\mapsto$ inconsistent set of theory atoms
- Inconsistent set of theory atoms $\mapsto$ cycle negative weight

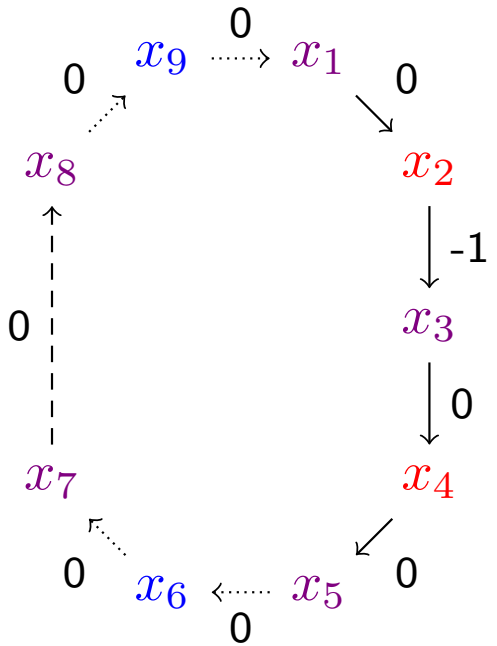
$\mathcal{I}$nteger $\mathcal{D}$ifference $\mathcal{L}$ogic:

- Atoms are inequalities of the form $x \leq y + c$
- Theory lemma $\mapsto$ inconsistent set of theory atoms
- Inconsistent set of theory atoms $\mapsto$ cycle negative weight



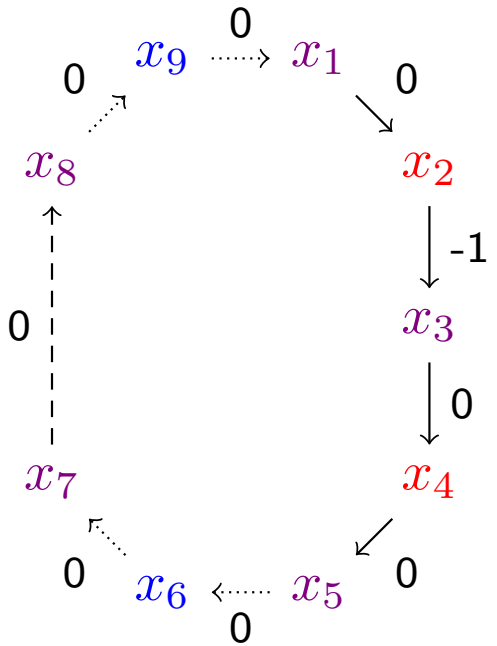
$\mathcal{I}$nteger $\mathcal{D}$ifference $\mathcal{L}$ogic:

- Atoms are inequalities of the form $x \leq y + c$
- Theory lemma $\mapsto$ inconsistent set of theory atoms
- Inconsistent set of theory atoms $\mapsto$ cycle negative weight
- $x \xrightarrow{c} y \equiv x \leq y + c$



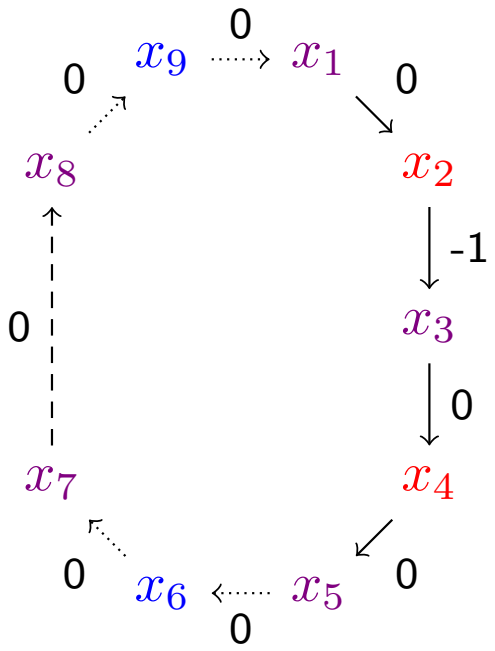
$\mathcal{I}$nteger $\mathcal{D}$ifference $\mathcal{L}$ogic:

- Atoms are inequalities of the form $x \leq y + c$
- Theory lemma $\mapsto$ inconsistent set of theory atoms
- Inconsistent set of theory atoms $\mapsto$ cycle negative weight
- $x \xrightarrow{c} y \equiv x \leq y + c$
- A, B, AB



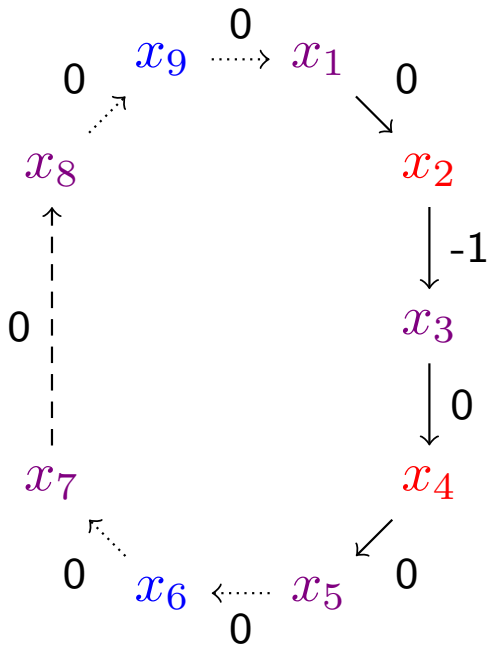
Interpolation in SMT

Theory interpolants in $\mathcal{IDL}$



Interpolation in SMT

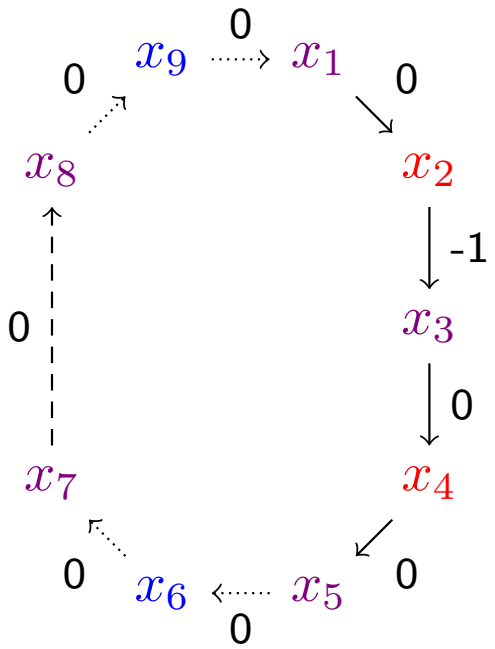
Theory interpolants in $\mathcal{IDL}$



- Interpolant $\equiv$ "summary" of A-edges [CGS10]

Interpolation in SMT

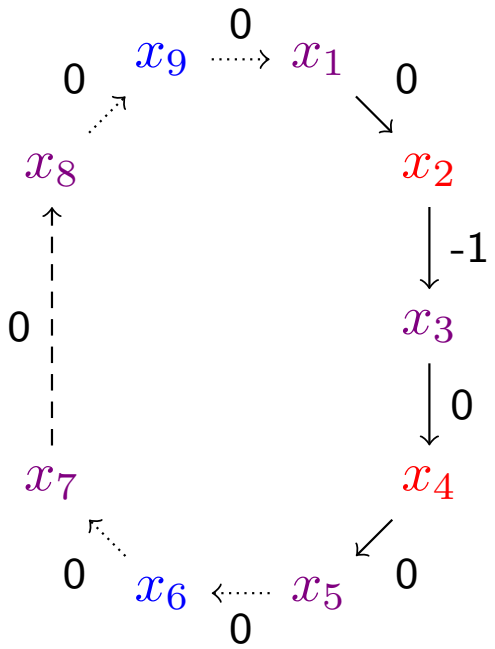
Theory interpolants in $\mathcal{IDL}$



- Interpolant $\equiv$ "summary" of A-edges [CGS10]
- Degree of summarization, handling of AB-edges

Interpolation in SMT

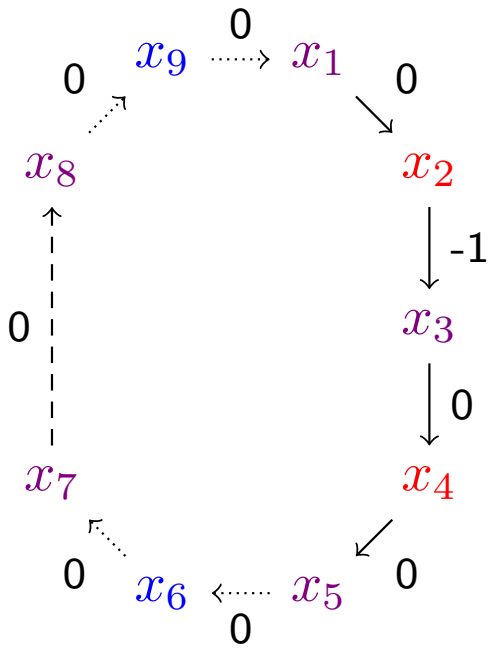
Theory interpolants in $\mathcal{IDL}$



- Interpolant $\equiv$ "summary" of A-edges [CGS10]
- Degree of summarization, handling of AB-edges
- Full, $x_7 \leq x_8$ to B: $x_1 \leq x_5 - 1$

Interpolation in SMT

Theory interpolants in $\mathcal{IDL}$



- Interpolant $\equiv$ "summary" of A-edges [CGS10]
- Degree of summarization, handling of AB-edges
- Full, $x_7 \leq x_8$ to B: $x_1 \leq x_5 - 1$
- Partial, $x_7 \leq x_8$ to A: $x_1 \leq x_3 - 1 \wedge x_3 \leq x_5 \wedge x_7 \leq x_8$

Path Interpolation

$\mathcal{IDL}$ Constraints

- Requirement $\tau_1 \rightarrow I_1$ $I_i \wedge \tau_{i+1} \rightarrow I_{i+1}$ $I_{n-1} \wedge \tau_n \rightarrow \bot$

Path Interpolation

$\mathcal{IDL}$ Constraints

- Requirement $\tau_1 \rightarrow I_1$ $I_i \wedge \tau_{i+1} \rightarrow I_{i+1}$ $I_{n-1} \wedge \tau_n \rightarrow \bot$
- Additional constraints on $\mathcal{IDL}$ theory interpolants
  - Degree of summarization
  - Handling of AB-edges

Path Interpolation

$\mathcal{IDL}$ Constraints

- Requirement $\tau_1 \rightarrow I_1$ $I_i \wedge \tau_{i+1} \rightarrow I_{i+1}$ $I_{n-1} \wedge \tau_n \rightarrow \bot$
- Additional constraints on $\mathcal{IDL}$ theory interpolants
  - Degree of summarization
  - Handling of AB-edges
- Labeled Interpolation Systems lifted to SMT

Path Interpolation

$\mathcal{IDL}$ Constraints

- Requirement $\tau_1 \rightarrow I_1$ $I_i \wedge \tau_{i+1} \rightarrow I_{i+1}$ $I_{n-1} \wedge \tau_n \rightarrow \bot$
- Additional constraints on $\mathcal{IDL}$ theory interpolants
  - Degree of summarization
  - Handling of AB-edges
- Labeled Interpolation Systems lifted to SMT
- Always satisfied for $L_1 \equiv \ldots \equiv L_n$

Theory interpolant manipulation

- Syntactic manipulation of $\mathcal{IDL}$ theory interpolants

Theory interpolant manipulation

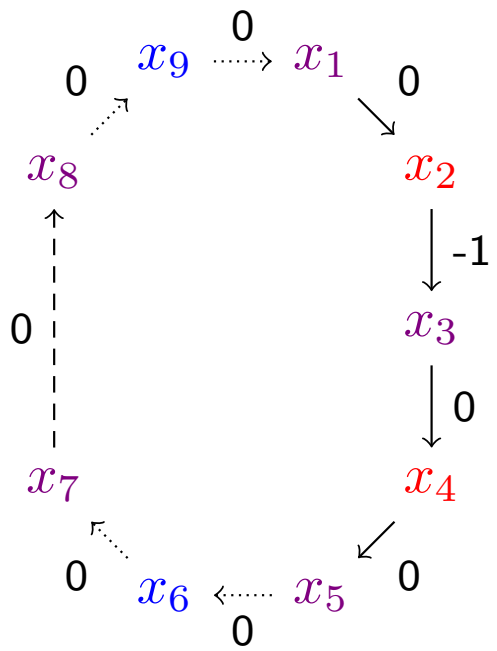
- Syntactic manipulation of $\mathcal{IDL}$ theory interpolants
- Deletion variables, terms

Theory interpolant manipulation

- Syntactic manipulation of $\mathcal{IDL}$ theory interpolants
- Deletion variables, terms
- Guided generation interpolants to help convergence

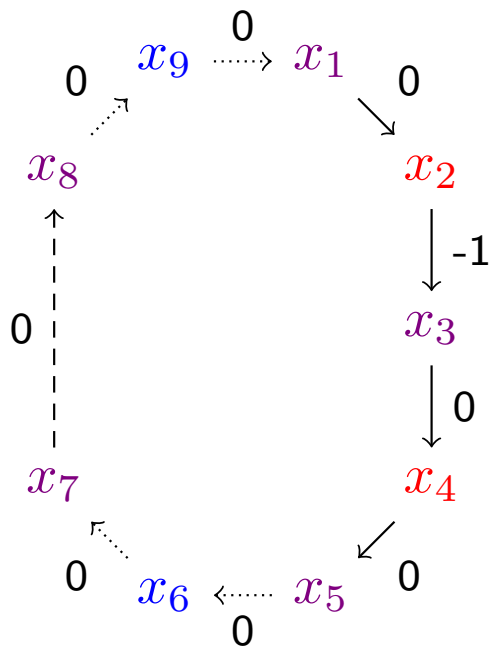
Theory interpolant manipulation

- Syntactic manipulation of $\mathcal{IDL}$ theory interpolants
- Deletion variables, terms
- Guided generation interpolants to help convergence



Theory interpolant manipulation

- Syntactic manipulation of $\mathcal{IDL}$ theory interpolants
- Deletion variables, terms
- Guided generation interpolants to help convergence



- $x_1 \leq x_3 - 1 \wedge x_3 \leq x_5 \wedge x_7 \leq x_8, \qquad x_1 \leq x_5 - 1$

Path Interpolation in SAFARI

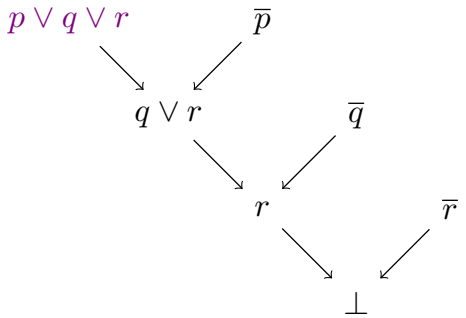
Challenges

- $A \wedge B$ is a conjunction of atoms

Path Interpolation in SAFARI

Challenges

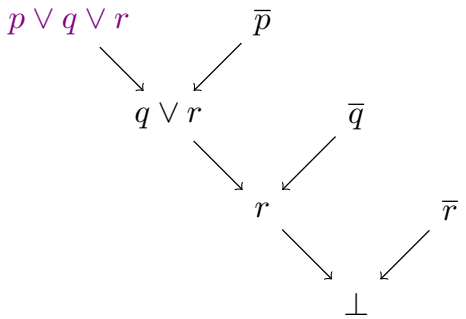
- $A \wedge B$ is a conjunction of atoms
- Linear proofs with one $\mathcal{IDL}$ theory lemma



Path Interpolation in SAFARI

Challenges

- $A \wedge B$ is a conjunction of atoms
- Linear proofs with one $\mathcal{IDL}$ theory lemma



- Very limited impact of interpolant manipulation

Term Abstraction:

Term Abstraction:

- input: (ϕ_1, ϕ_2), $T = \{t_1, \ldots, t_n\}$

Term Abstraction:

- input: (ϕ_1, ϕ_2), $T = \{t_1, \ldots, t_n\}$
- For every $t_i \in T$ check if $\phi_1(c_i/t_i) \wedge \phi_2(d_i/t_i)$ is still inconsistent

Term Abstraction:

- input: (ϕ_1, ϕ_2), $T = \{t_1, \ldots, t_n\}$
- For every $t_i \in T$ check if $\phi_1(c_i/t_i) \wedge \phi_2(d_i/t_i)$ is still inconsistent
- return $\hat{\phi}_1, \hat{\phi}_2$

Heuristics

CM + TA

Benchmark	Result	No Opt.	TA + CM		Benchmark	Result	No Opt.	TA + CM
binarySort	safe	100	21		p08	safe	100	3
bubbleSort	safe	100	5		p08Test	safe	100	2
compare	safe	100	2		p09	safe	100	3
copy	safe	100	2		p09Test	safe	100	2
copyTest	safe	100	21		p11	unsafe	0	0
filter	safe	100	0		p11Test	unsafe	0	0
filterTest	safe	0	0		p13	unsafe	0	0
find	safe	100	4		p15	unsafe	4	8
findTest	safe	100	29		p15Test	unsafe	3	6
heapAsArray	safe	31	3		p18	unsafe	0	0
init	safe	100	2		p19	unsafe	0	0
initTest	safe	100	11		p19Test	unsafe	0	0
insertionSort	safe	8	5		p20	safe	100	3
maxInArray	safe	100	11		p20Test	safe	100	2
maxInArrayTest	safe	100	7		p21	safe	3	2
p01	safe	100	3		p22	safe	3	2
p01Test	safe	100	2		partInit	safe	100	5
p03	safe	100	3		partition	safe	80	6
p03Test	safe	100	2		partitionTest	safe	100	22
p04	unsafe	0	0		running	safe	100	3
p04Test	unsafe	0	0		vararg	safe	100	2
p06	unsafe	100	2					

Heuristics

CM + TA

Benchmark	Result	Time (s)		Benchmark	Result	Time (s)
binarySort	safe	10.92		p08	safe	0.62
bubbleSort	safe	3.65		p08Test	safe	0.41
compare	safe	0.55		p09	safe	0.75
copy	safe	0.42		p09Test	safe	0.49
copyTest	safe	9.17		p11	unsafe	0.31
filter	safe	0.82		p11Test	unsafe	0.26
filterTest	safe	0.73		p13	unsafe	0.52
find	safe	0.78		p15	unsafe	3.43
findTest	safe	9.93		p15Test	unsafe	2.59
heapAsArray	safe	0.74		p18	unsafe	0.17
init	safe	0.45		p19	unsafe	0.17
initTest	safe	4.22		p19Test	unsafe	0.19
insertionSort	safe	2.07		p20	safe	0.72
maxInArray	safe	4.70		p20Test	safe	0.41
maxInArrayTest	safe	2.34		p21	safe	0.48
p01	safe	0.70		p22	safe	0.51
p01Test	safe	0.45		partInit	safe	1.69
p03	safe	0.72		partition	safe	1.79
p03Test	safe	0.47		partitionTest	safe	18.03
p04	unsafe	0.29		running	safe	0.48
p04Test	unsafe	0.23		vararg	safe	0.46
p06	unsafe	0.54				

Conclusions

- (at least) Two dimensions for driving refinement procedures

Conclusions

- (at least) Two dimensions for driving refinement procedures
  - Which unsat core?
    - CM preserves at most the labeled unwinding

Conclusions

- (at least) Two dimensions for driving refinement procedures
  - Which unsat core?
    - CM preserves at most the labeled unwinding
  - Which set of interpolants?
    - Still not clear how to play with different labelings
    - Term abstraction tries to "lift" counterexamples in order to generalize them

Conclusions

- (at least) Two dimensions for driving refinement procedures
  - Which unsat core?
    - CM preserves at most the labeled unwinding
  - Which set of interpolants?
    - Still not clear how to play with different labelings
    - Term abstraction tries to "lift" counterexamples in order to generalize them

$\Rightarrow$ Which parameters do we need in order to drive interpolation-based refinement procedures?

Thank you! Questions?

References I

Francesco Alberti, Roberto Bruttomesso, Silvio Ghilardi, Silvio Ranise, and Natasha Sharygina.
Lazy abstraction with interpolants for arrays.
In Nikolaj Bjørner and Andrei Voronkov, editors, *LPAR*, volume 7180 of *Lecture Notes in Computer Science*, pages 46–61. Springer, 2012.

Francesco Alberti, Roberto Bruttomesso, Silvio Ghilardi, Silvio Ranise, and Natasha Sharygina.
Safari: Smt-based abstraction for arrays with interpolants.
In Madhusudan and Seshia [MS12], pages 679–685.

Alessandro Cimatti, Alberto Griggio, and Roberto Sebastiani.
Efficient generation of craig interpolants in satisfiability modulo theories.
ACM Trans. Comput. Log., 12(1):7, 2010.

References II

Vijay D'Silva, Daniel Kroening, Mitra Purandare, and Georg Weissenbacher.
Interpolant strength.
In Gilles Barthe and Manuel V. Hermenegildo, editors, *VMCAI*, volume 5944 of *Lecture Notes in Computer Science*, pages 129–145. Springer, 2010.

Arie Gurfinkel, Simone Fulvio Rollini, and Natasha Sharygina.
Propositional interpolation systems for model checking.
In *ATVA*, 2013.
To appear.

References III


Thomas A. Henzinger, Ranjit Jhala, Rupak Majumdar, and Kenneth L. McMillan.
Abstractions from proofs.
In Neil D. Jones and Xavier Leroy, editors, *POPL*, pages 232–244. ACM, 2004.


T. A. Henzinger, R. Jhala, R. Majumdar, and G. Sutre.
Lazy Abstraction.
In *POPL*, 2002.


Kenneth L. McMillan.
Applications of craig interpolation to model checking.
In Jerzy Marcinkowski and Andrzej Tarlecki, editors, *CSL*, volume 3210 of *Lecture Notes in Computer Science*, pages 22–23. Springer, 2004.

References IV

Kenneth L. McMillan.
Lazy abstraction with interpolants.
In Thomas Ball and Robert B. Jones, editors, *CAV*, volume 4144 of *Lecture Notes in Computer Science*, pages 123–136. Springer, 2006.

P. Madhusudan and Sanjit A. Seshia, editors.
Computer Aided Verification - 24th International Conference, CAV 2012, Berkeley, CA, USA, July 7-13, 2012 Proceedings, volume 7358 of *Lecture Notes in Computer Science*. Springer, 2012.

Pavel Pudlák.
Lower bounds for resolution and cutting plane proofs and monotone computations.
J. Symb. Log., 62(3):981–998, 1997.

References V

Simone Fulvio Rollini, Ondrej Sery, and Natasha Sharygina.
Leveraging interpolant strength in model checking.
In Madhusudan and Seshia [MS12], pages 193–209.

Greta Yorsh and Madanlal Musuvathi.
A combination method for generating interpolants.
In Robert Nieuwenhuis, editor, *CADE*, volume 3632 of *Lecture Notes in Computer Science*, pages 353–368. Springer, 2005.