Synthesis of ranking functions using extremal counterexamples

Laure Gonnord, David Monniaux, Gabriel Radanne
(Lucas Seguinot)
Slides : credits Gabriel Radanne

Lyon1/LIP

2 dec 2014 - Compsys WG

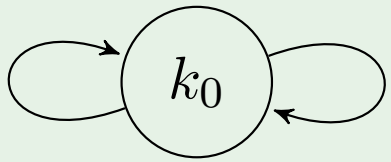
Why ?

Our goal :

- Prove termination of **some** sequential programs.
- with a scalable algorithm

Counter Automata

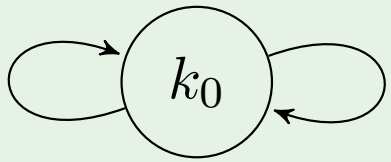
$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}}$$



$$\boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$

Counter Automata

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}}$$



$$\boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$

The Initial position

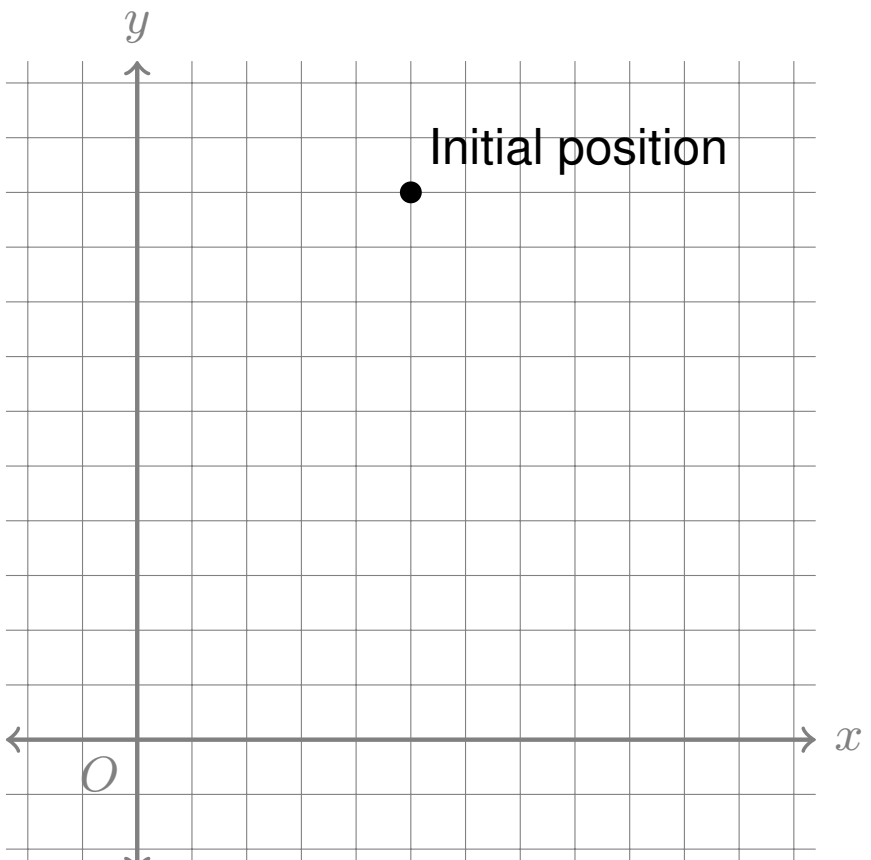
$x = 5$ and $y = 10$

Counter Automata

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$

The Initial position

$x = 5$ and $y = 10$

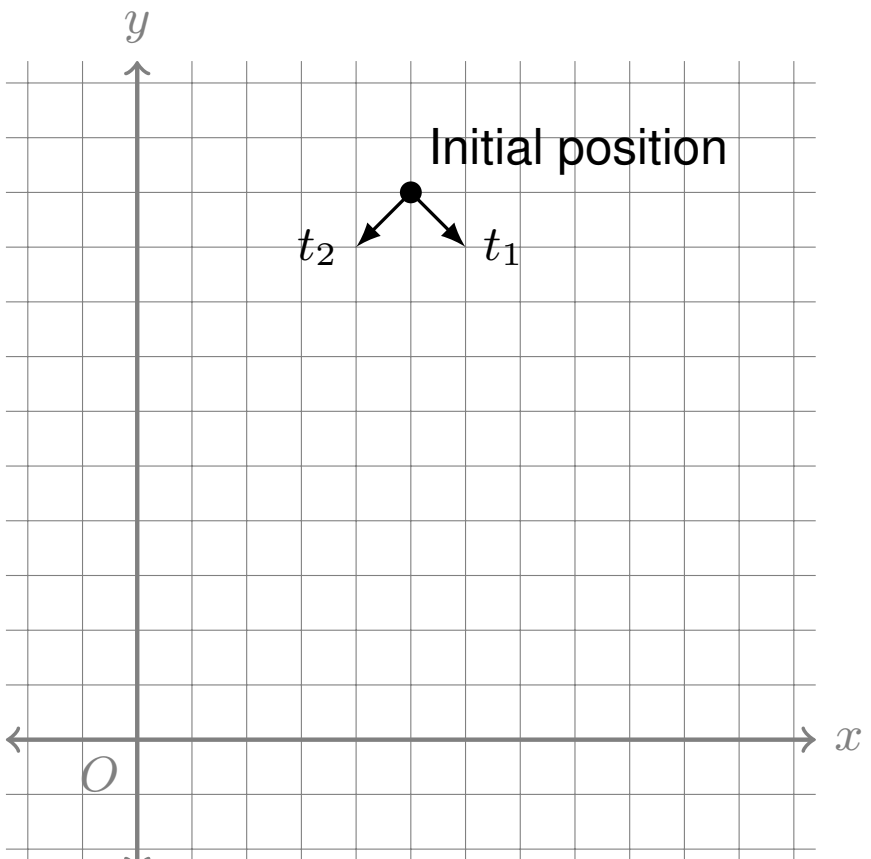


Counter Automata

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$

The Initial position

$x = 5$ and $y = 10$



Counter Automata

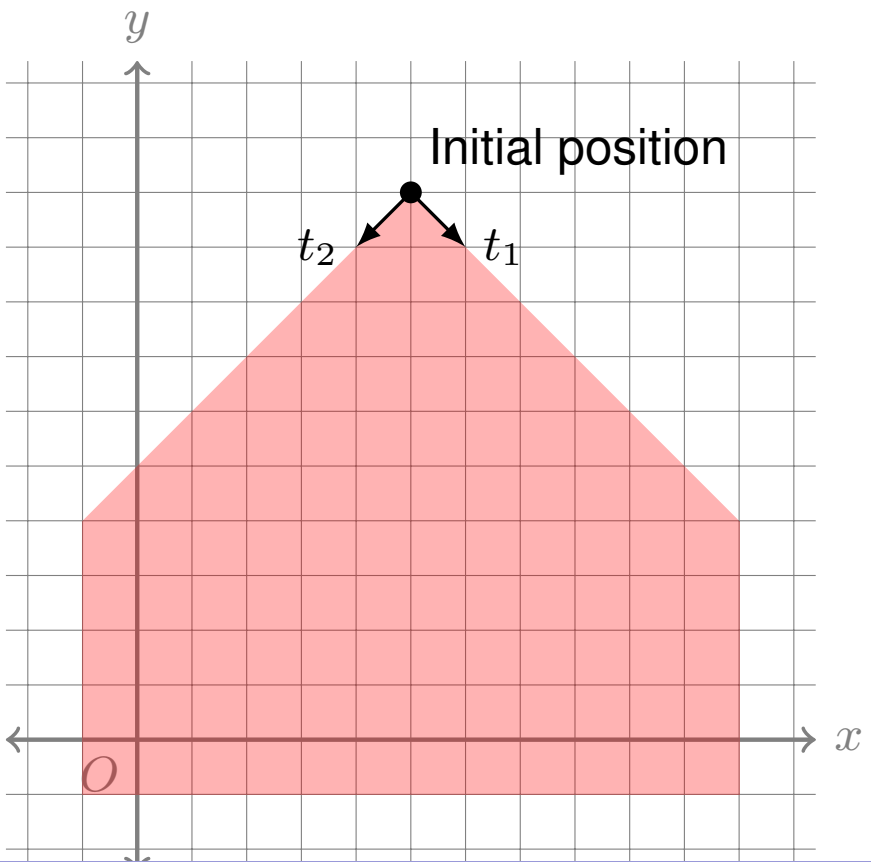
$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$

The Initial position

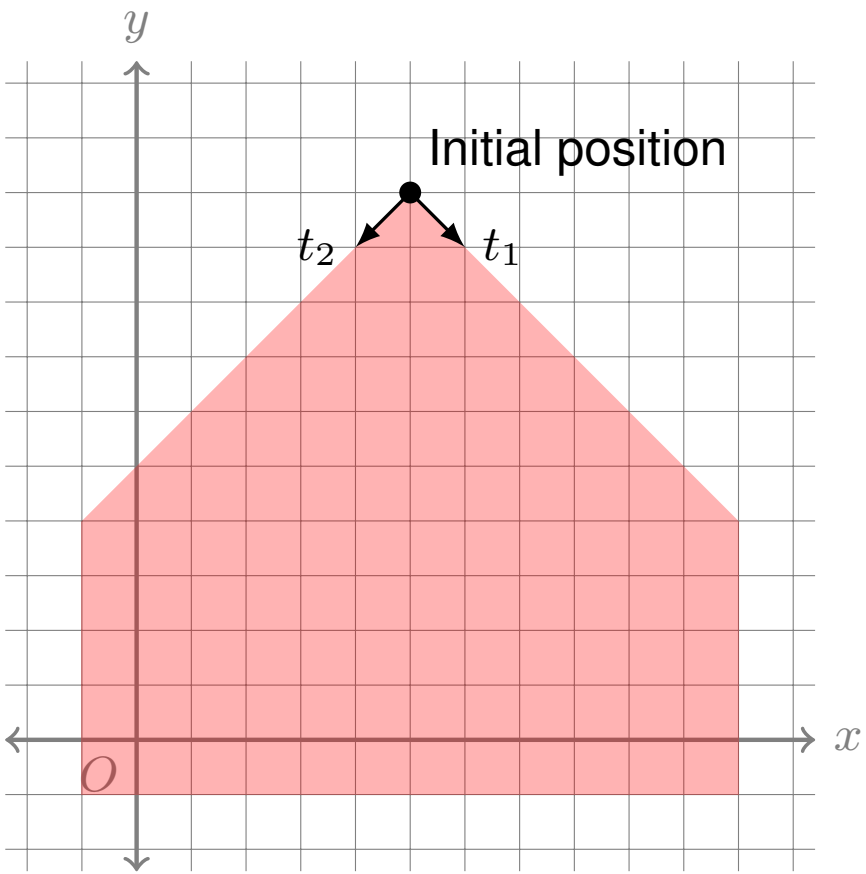
$x = 5$ and $y = 10$

The Invariants (given by ASPIC)

$$0 \leqslant x+1 \quad x \leqslant 11$$
$$0 \leqslant y+1 \quad y \leqslant x+5$$
$$x+y \leqslant 15$$



Invariant representation



Here
$\mathcal{I} = \{0 \leqslant x+1, x \leqslant 11, 0 \leqslant y+1, y \leqslant x+5, x+y \leqslant 15\}$,

i.e. $\mathcal{I} = \{\boldsymbol{x} \mid \boldsymbol{a}_i \cdot \boldsymbol{x} + b_i \geq 0\}$
with

$$\boldsymbol{a} = \begin{pmatrix}1\\0\end{pmatrix} \begin{pmatrix}-1\\0\end{pmatrix} \begin{pmatrix}0\\1\end{pmatrix} \begin{pmatrix}1\\-1\end{pmatrix} \begin{pmatrix}1\\1\end{pmatrix}$$
$$\boldsymbol{b} = \quad 1 \quad\quad 11 \quad\quad 1 \quad\quad 5 \quad\quad 15$$

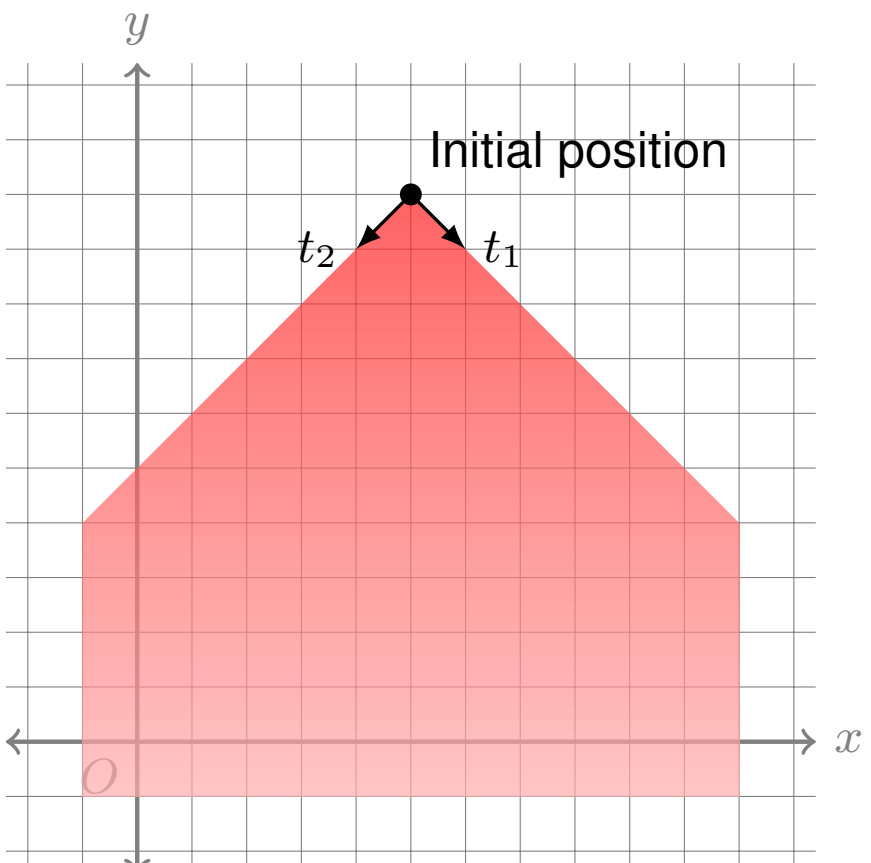
The Automaton

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$

A linear ranking function

$$\rho(x, y) = y + 1$$

- Linear
- Decreasing
- Positive



Proving termination

Linear ranking function

- Decreasing by at least one along the transitions
- Positive
- Linear

Weak linear ranking function

- Decreasing along the transitions
- Positive
- Linear

Note

The null function is always a weak linear ranking function.

Algorithm to synthesize a ranking function

Two main ideas

- Big block encoding.
- Treat the loops globally.

First Goal

- Find a linear ranking function.
- In programs with one control point.
- A “maximally strict” one.

Some Maths

Input

- τ a transition relation
- $\mathcal{I}$ an invariant $\mathcal{I} = \left\{ \boldsymbol{x} \;\middle|\; \boldsymbol{a}_i \cdot \boldsymbol{x} + b_i \geq 0 \right\}$

Looking for $\rho = \lambda \boldsymbol{x} + \ell$

- Positive on $\mathcal{I}$: $\rho(\boldsymbol{x}) = (\sum_{i=1}^{m} \lambda_i \boldsymbol{a}_i) \cdot \boldsymbol{x} + \ell$ (Farkas)
- Decreasing : $\forall (\boldsymbol{x}, \boldsymbol{x}') \in \ldots, \lambda.(\boldsymbol{x} - \boldsymbol{x}') \geqslant 0$

The second condition is $\lambda.u \geqslant 0$, $\forall u$ in a certain polyhedron :

$$\mathcal{P}_{\mathcal{I},\tau} = \{\boldsymbol{x} - \boldsymbol{x}' | \boldsymbol{x} \in \mathcal{I} \text{ and } (\boldsymbol{x}, \boldsymbol{x}') \in \tau\},$$

or **for each of its generators $\boldsymbol{u}_j$.**

Simple algorithm for one control point

An incremental algorithm

1. Consider ρ a (weak) ranking function.
2. Find a counterexample.
   i.e. an element $\boldsymbol{u} \in \mathcal{P}_{\mathcal{I},\tau} = \{\boldsymbol{x} - \boldsymbol{x'} | \ \boldsymbol{x} \in \mathcal{I}, (\boldsymbol{x}, \boldsymbol{x'}) \in \tau\}$
   which contradicts that ρ is a strict ranking function.
   $\rightarrow$ If none is found, the current ranking function is strict.
   Stop.
3. Add $\boldsymbol{u}$ to a set $\mathcal{C}$.
4. Use $\mathcal{C}$ to compute a new "maximally strict" ranking function.
5. Go back to step 2.

Simple algorithm for one control point

An incremental algorithm

1. Consider ρ a (weak) ranking function.
2. Find a counterexample.
   i.e. an element $\boldsymbol{u} \in \mathcal{P}_{\mathcal{I},\tau} = \{\boldsymbol{x} - \boldsymbol{x'} | \ \boldsymbol{x} \in \mathcal{I}, (\boldsymbol{x}, \boldsymbol{x'}) \in \tau\}$
   which contradicts that ρ is a strict ranking function.
   $\rightarrow$ If none is found, the current ranking function is strict. Stop.
3. Add $\boldsymbol{u}$ to a set $\mathcal{C}$.
4. Use $\mathcal{C}$ to compute a new "maximally strict" ranking function.
5. Go back to step 2.

Simple algorithm for one control point, 4.

$$\rho(\boldsymbol{x}) = \left(\sum_{i=1}^{m} \lambda_i \boldsymbol{a}_i\right) \cdot \boldsymbol{x} + \sum_{i=1}^{m} \lambda_i b_i \quad \text{with } \mathcal{I} = \left\{\boldsymbol{x} \;\middle|\; \boldsymbol{a}_i \cdot \boldsymbol{x} + b_i \geq 0\right\}$$

Simple algorithm for one control point, 4.

$$\rho(\boldsymbol{x}) = \left(\sum_{i=1}^{m} \lambda_i \boldsymbol{a}_i\right) \cdot \boldsymbol{x} + \sum_{i=1}^{m} \lambda_i b_i \quad \text{with } \mathcal{I} = \left\{\boldsymbol{x} \;\middle|\; \boldsymbol{a}_i \cdot \boldsymbol{x} + b_i \geq 0\right\}$$

Definition : $LP(\mathcal{C}, \mathcal{I})$

- $\mathcal{C} = (\boldsymbol{u}_1, \ldots \boldsymbol{u}_N)$ a set of generators of the polyhedron $\mathcal{P}_{\mathcal{I},\tau}$,

$$LP(\mathcal{C}, \mathcal{I}) = \begin{cases} Maximize \sum_i \delta_i \text{ s.t.} & \\ \lambda_1, \ldots, \lambda_m \geqslant 0 & \\ 0 \leqslant \delta_j \leqslant 1 & \text{for all } 1 \leqslant j \leqslant N \\ \sum_{i=1}^{m} \lambda_i (\boldsymbol{u}_j . \boldsymbol{a}_i) \geqslant \delta_j & \text{for all } 1 \leqslant j \leqslant N \end{cases}$$

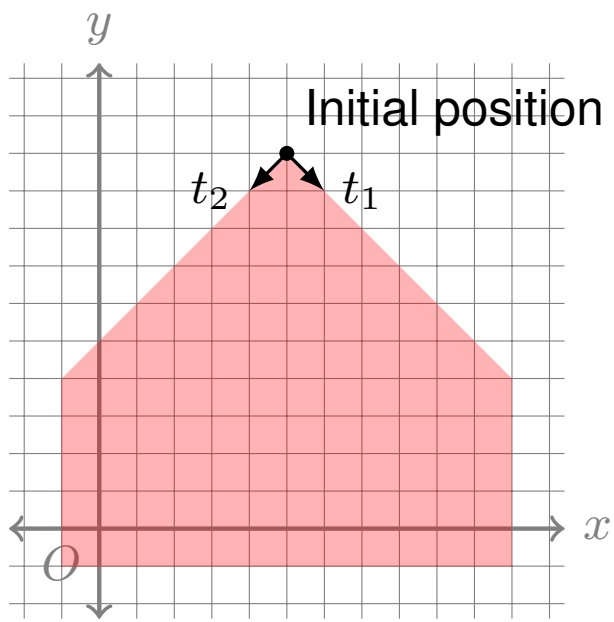
Proposition

- $\boldsymbol{\lambda} = \mathbf{0}$ is always a solution.
- ρ is “maximally strict” on $\mathcal{C}$.

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x - 1 \\ y := y - 1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x + 1 \\ y := y - 1 \end{array}}$$

Current State

$$\rho(\begin{pmatrix} x \\ y \end{pmatrix}) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \end{pmatrix} + 0,\ \mathcal{C} = \{\}$$



Give the SMT-solver the constraint :

$$\mathcal{I} \wedge \tau \wedge \rho(\begin{pmatrix} x \\ y \end{pmatrix}) - \rho(\begin{pmatrix} x' \\ y' \end{pmatrix}) \leqslant 0$$

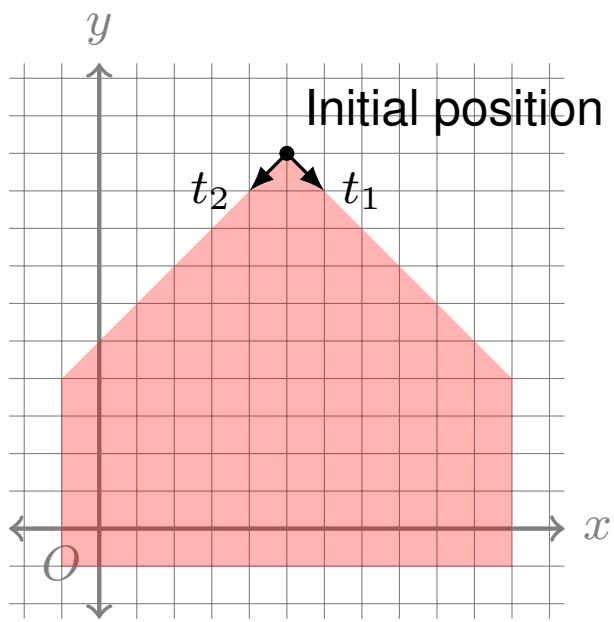
$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x - 1 \\ y := y - 1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x + 1 \\ y := y - 1 \end{array}}$$

Current State

$$\rho(\begin{pmatrix} x \\ y \end{pmatrix}) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \end{pmatrix} + 0,\ \mathcal{C} = \{\}$$

Give the SMT-solver the constraint :

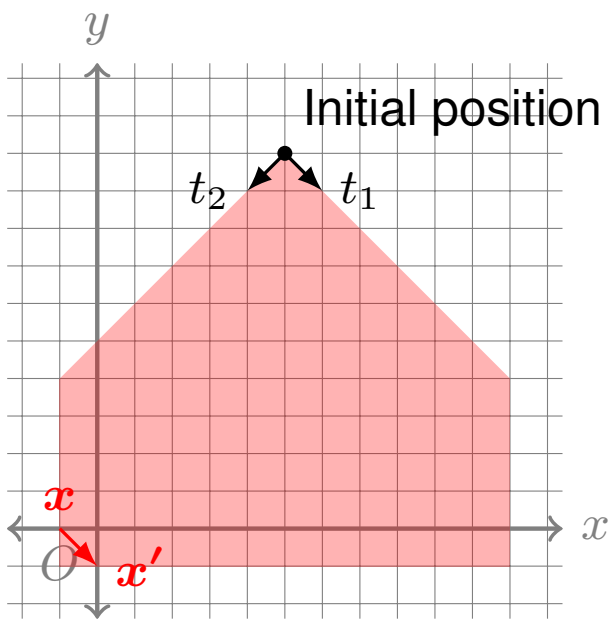
$$\mathcal{I} \wedge \tau \wedge \begin{pmatrix} x - x' \\ y - y' \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \end{pmatrix} \leqslant 0$$



$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x - 1 \\ y := y - 1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x + 1 \\ y := y - 1 \end{array}}$$

Current State

$$\rho(\begin{pmatrix} x \\ y \end{pmatrix}) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \end{pmatrix} + 0,\ \mathcal{C} = \{\}$$



Give the SMT-solver the constraint :

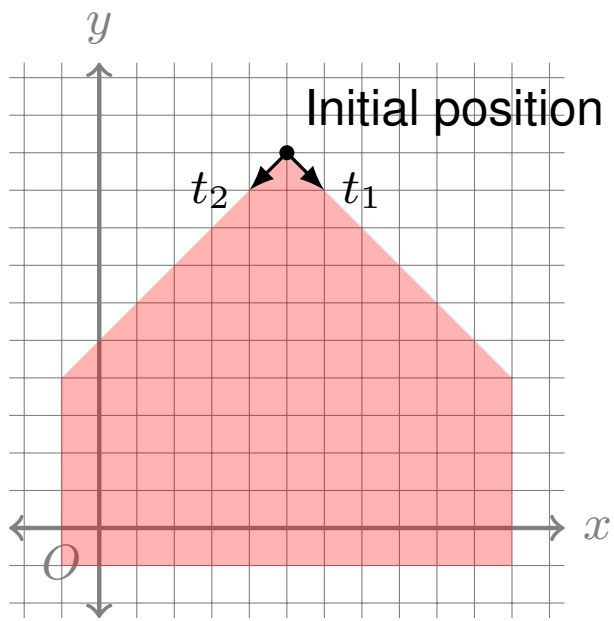
$$\mathcal{I} \wedge \tau \wedge \begin{pmatrix} x - x' \\ y - y' \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \end{pmatrix} \leqslant 0$$

$$\begin{array}{ll} x = -1 & x' = 0 \\ y = 0 & y' = -1 \end{array} \qquad \boldsymbol{u} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x - 1 \\ y := y - 1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x + 1 \\ y := y - 1 \end{array}}$$

Current State

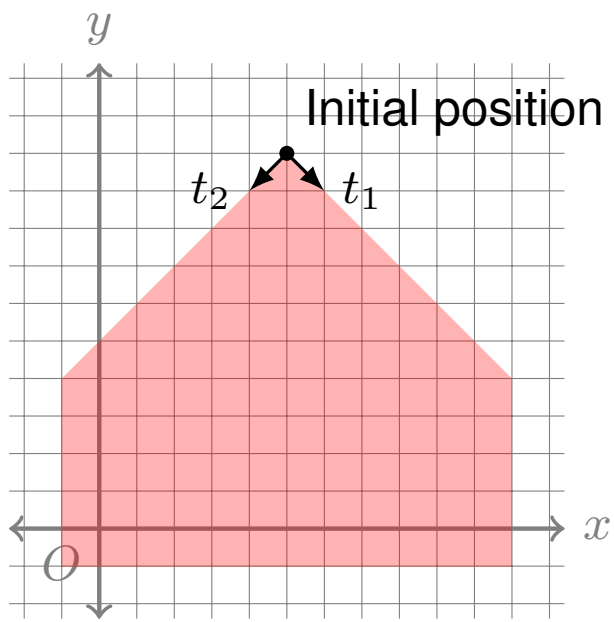
$$\rho\left(\begin{pmatrix} x \\ y \end{pmatrix}\right) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \end{pmatrix} + 0, \; \mathcal{C} = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix} \right\}$$



$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$

Current State

$$\rho\left(\begin{pmatrix} x \\ y \end{pmatrix}\right) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \end{pmatrix} + 0,\ \mathcal{C} = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix} \right\}$$



Give to the LP-solver the problem :

$$\begin{cases} \textit{Maximize } \delta_1 \text{ s.t.} \\ \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5 \geqslant 0 \\ 0 \leqslant \delta_1 \leqslant 1 \\ -\lambda_1 + \lambda_2 + \lambda_3 - 2\lambda_4 \geqslant \delta_1 \end{cases}$$

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{l} x := x-1 \\ y := y-1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{l} x := x+1 \\ y := y-1 \end{array}}$$

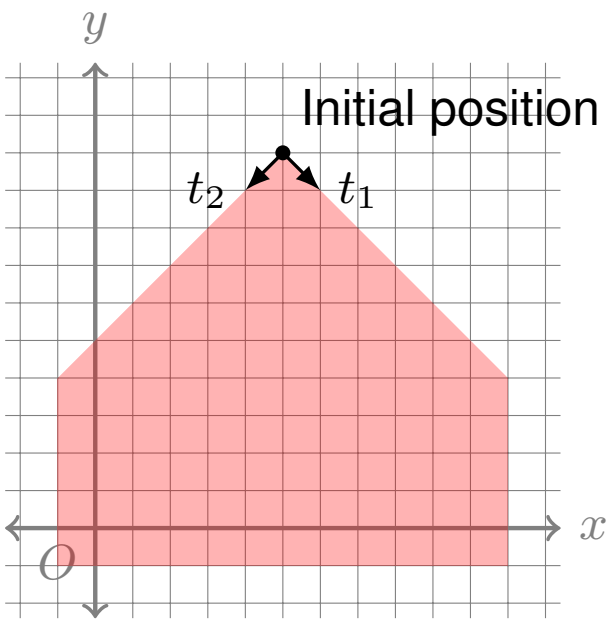
Current State

$$\rho\left(\begin{pmatrix} x \\ y \end{pmatrix}\right) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \end{pmatrix} + 0,\ \mathcal{C} = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix} \right\}$$

$\rightarrow$ Gives back

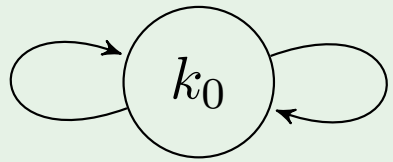
$\lambda_2 = 1, \lambda_1 = \lambda_3 = \lambda_4 = \lambda_5 = 0.$

Then $\boldsymbol{l} = \boldsymbol{a}_2 = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$ and $\ell = \boldsymbol{b}_2 = 11.$



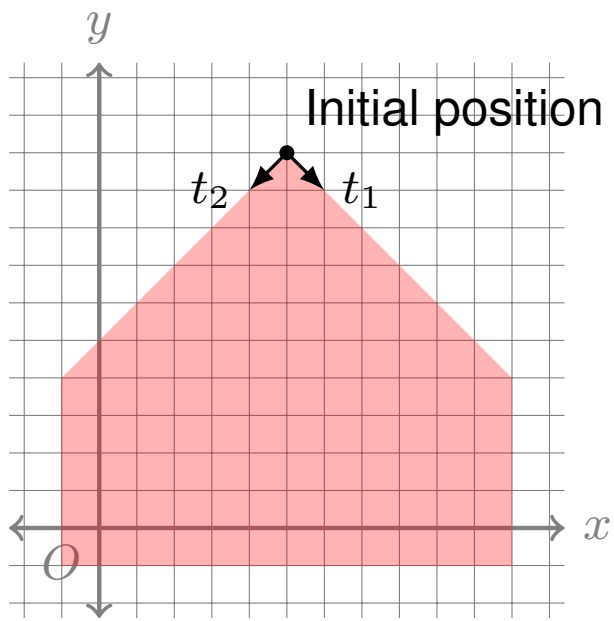
$$\rho\left(\begin{pmatrix} x \\ y \end{pmatrix}\right) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} -1 \\ 0 \end{pmatrix} + 11$$

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}} \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$

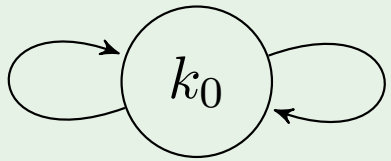


Current State

$$\rho\left(\begin{pmatrix} x \\ y \end{pmatrix}\right) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} -1 \\ 0 \end{pmatrix} + 11, \; \mathcal{C} = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix} \right\}$$

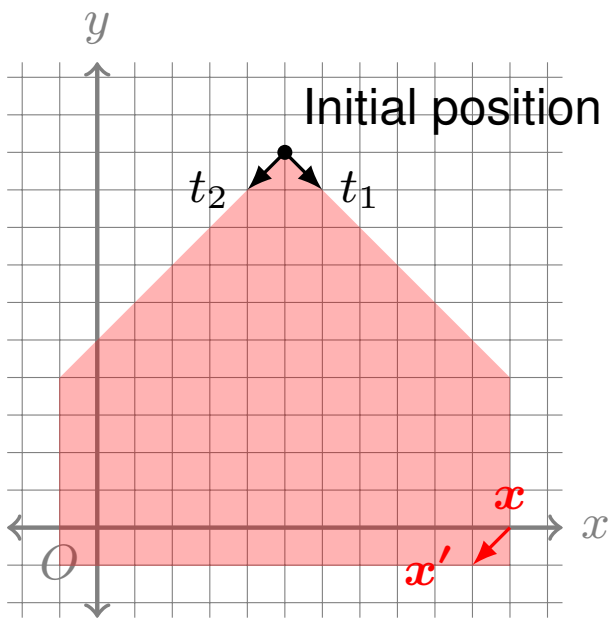


$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}} \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$



Current State

$$\rho\left(\begin{pmatrix} x \\ y \end{pmatrix}\right) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} -1 \\ 0 \end{pmatrix} + 11, \; \mathcal{C} = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix} \right\}$$



Give the SMT-solver the constraint :

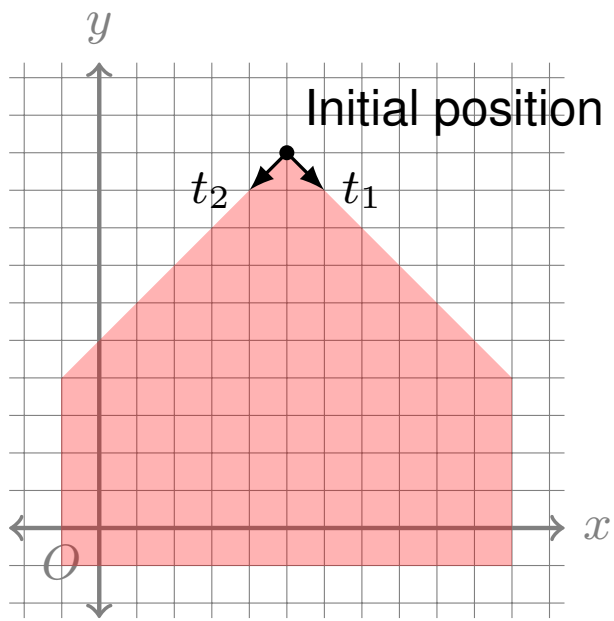
$$\mathcal{I} \wedge \tau \wedge \begin{pmatrix} x - x' \\ y - y' \end{pmatrix} \cdot \begin{pmatrix} -1 \\ 0 \end{pmatrix} \leqslant 0$$

$$\begin{array}{ll} x = 11 & x' = 10 \\ y = 0 & y' = -1 \end{array} \qquad \boldsymbol{u} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x - 1 \\ y := y - 1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x + 1 \\ y := y - 1 \end{array}}$$

Current State

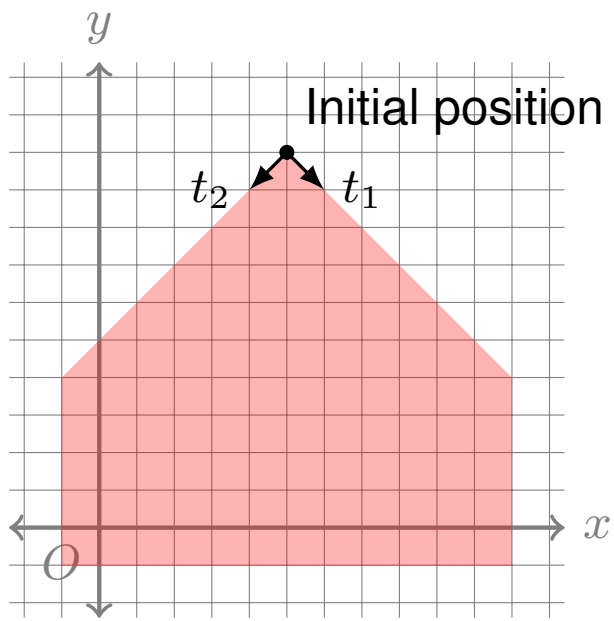
$$\rho(\begin{pmatrix} x \\ y \end{pmatrix}) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} -1 \\ 0 \end{pmatrix} + 11,\ \mathcal{C} = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}$$



$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$

Current State

$$\rho\left(\begin{pmatrix} x \\ y \end{pmatrix}\right) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} -1 \\ 0 \end{pmatrix} + 11,\ \mathcal{C} = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}$$



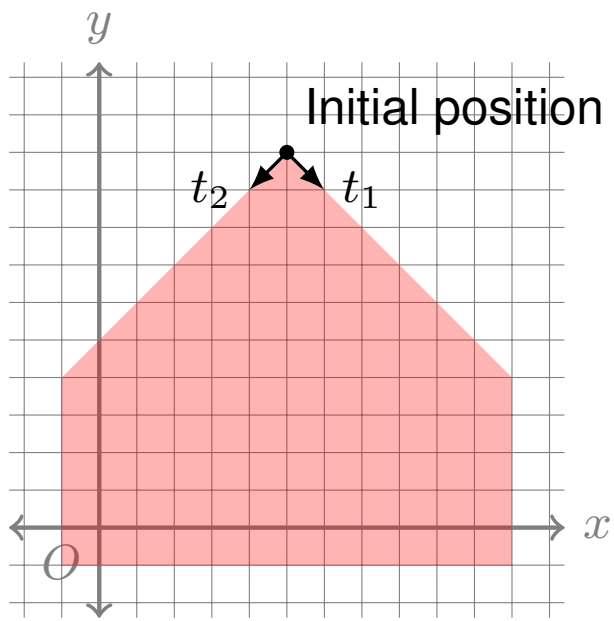
Give to the LP-solver the problem :

$$\begin{cases} \textit{Maximize } \delta_1 + \delta_2 \text{ s.t.} \\ \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5 \geqslant 0 \\ 0 \leqslant \delta_1, \delta_2 \leqslant 1 \\ -\lambda_1 + \lambda_2 + \lambda_3 - 2\lambda_4 \geqslant \delta_1 \\ \lambda_1 - \lambda_2 + \lambda_3 - 2\lambda_5 \geqslant \delta_2 \end{cases}$$

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$

Current State

$$\rho\left(\begin{pmatrix} x \\ y \end{pmatrix}\right) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} -1 \\ 0 \end{pmatrix} + 11,\ \mathcal{C} = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}$$



$\rightarrow$ Gives back

$\lambda_3 = 1, \lambda_1 = \lambda_2 = \lambda_4 = \lambda_5 = 0.$

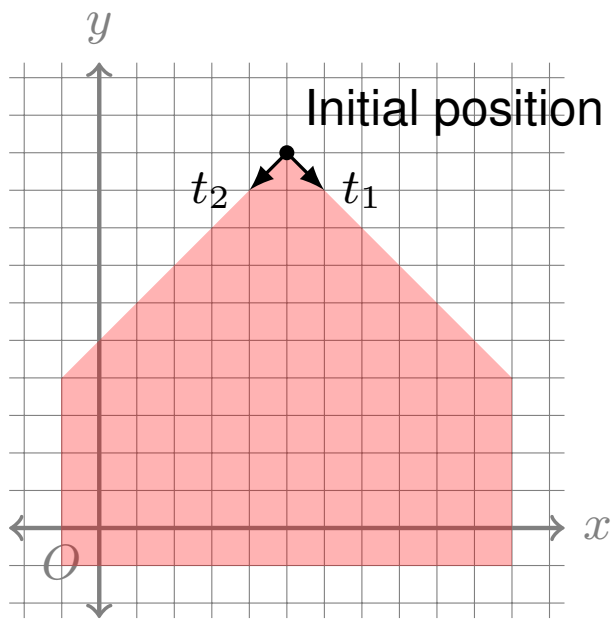
Then $\boldsymbol{l} = \boldsymbol{a}_3 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and $\ell = \boldsymbol{b}_3 = 1.$

$$\rho\left(\begin{pmatrix} x \\ y \end{pmatrix}\right) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} + 1$$

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x - 1 \\ y := y - 1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x + 1 \\ y := y - 1 \end{array}}$$

Current State

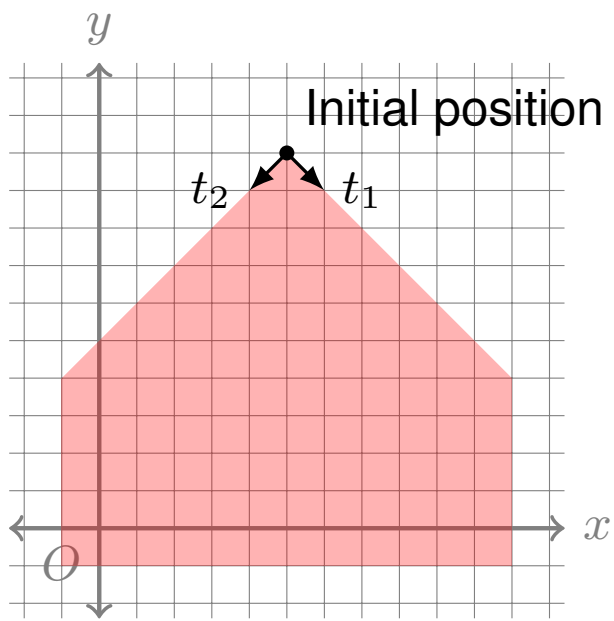
$$\rho(\begin{pmatrix} x \\ y \end{pmatrix}) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} + 1,\ \mathcal{C} = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}$$



$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x - 1 \\ y := y - 1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x + 1 \\ y := y - 1 \end{array}}$$

Current State

$$\rho(\begin{pmatrix} x \\ y \end{pmatrix}) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} + 1,\ \mathcal{C} = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}$$



Give the SMT-solver the constraint :

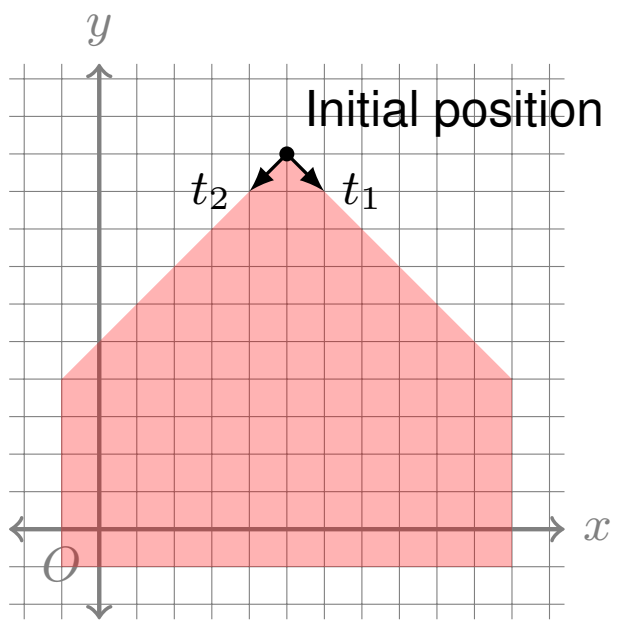
$$\mathcal{I} \wedge \tau \wedge \begin{pmatrix} x - x' \\ y - y' \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} \leqslant 0$$

which is **unsat**.

$$\boldsymbol{t_2}: \frac{0 \leqslant x \wedge 0 \leqslant y}{\begin{array}{c} x := x-1 \\ y := y-1 \end{array}} \qquad k_0 \qquad \boldsymbol{t_1}: \frac{x \leqslant 10 \wedge 0 \leqslant y}{\begin{array}{c} x := x+1 \\ y := y-1 \end{array}}$$

Current State

$$\rho\left(\begin{pmatrix} x \\ y \end{pmatrix}\right) = \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} + 1,\ \mathcal{C} = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}$$



Output

$$\rho(x, y) = y + 1$$

ρ is a **strict** ranking function.

Limitation 1

This algorithm doesn't terminate in general

- The set of counter examples is infinite.
- If there is no strict ranking function.

Fix : limit the search area for u

- impose counter examples to be in the boundary of $\mathcal{P}_{\mathcal{I},\tau}$ (max-smt).
- look for $u \notin span(previous\_u)$

Limitation 2

This algorithm only computes rankings of dim 1

- But all programs are not linear !
- Compute **lexicographic** linear ranking functions (in $\mathbb{Q}^m$).

We use the same greedy algorithm as in [Alias et al, SAS 2010]

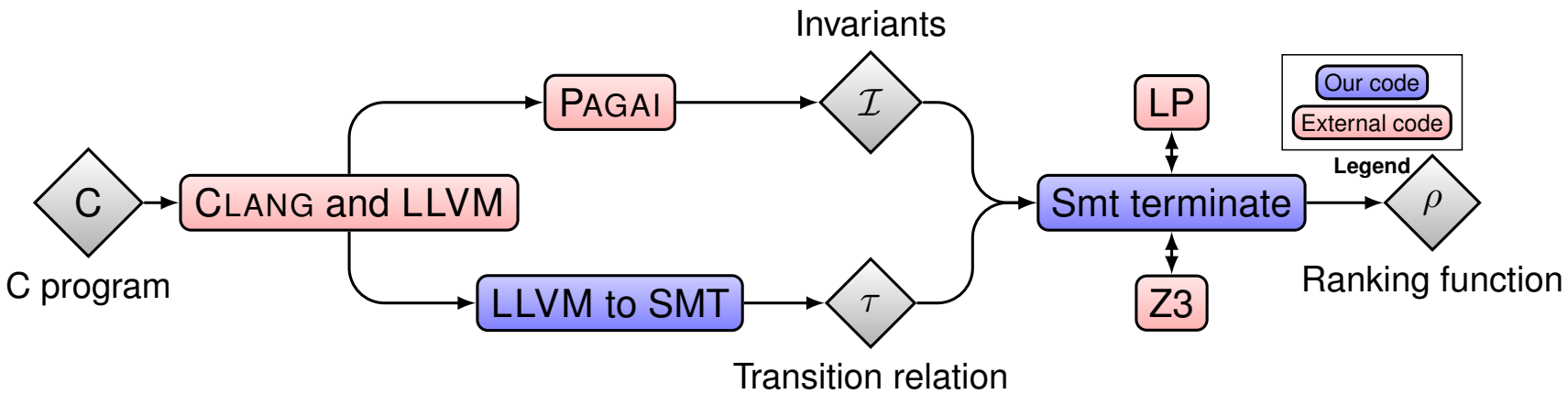
Limitation 3

This algorithm is only for one control point

- But not all programs have only one control point !

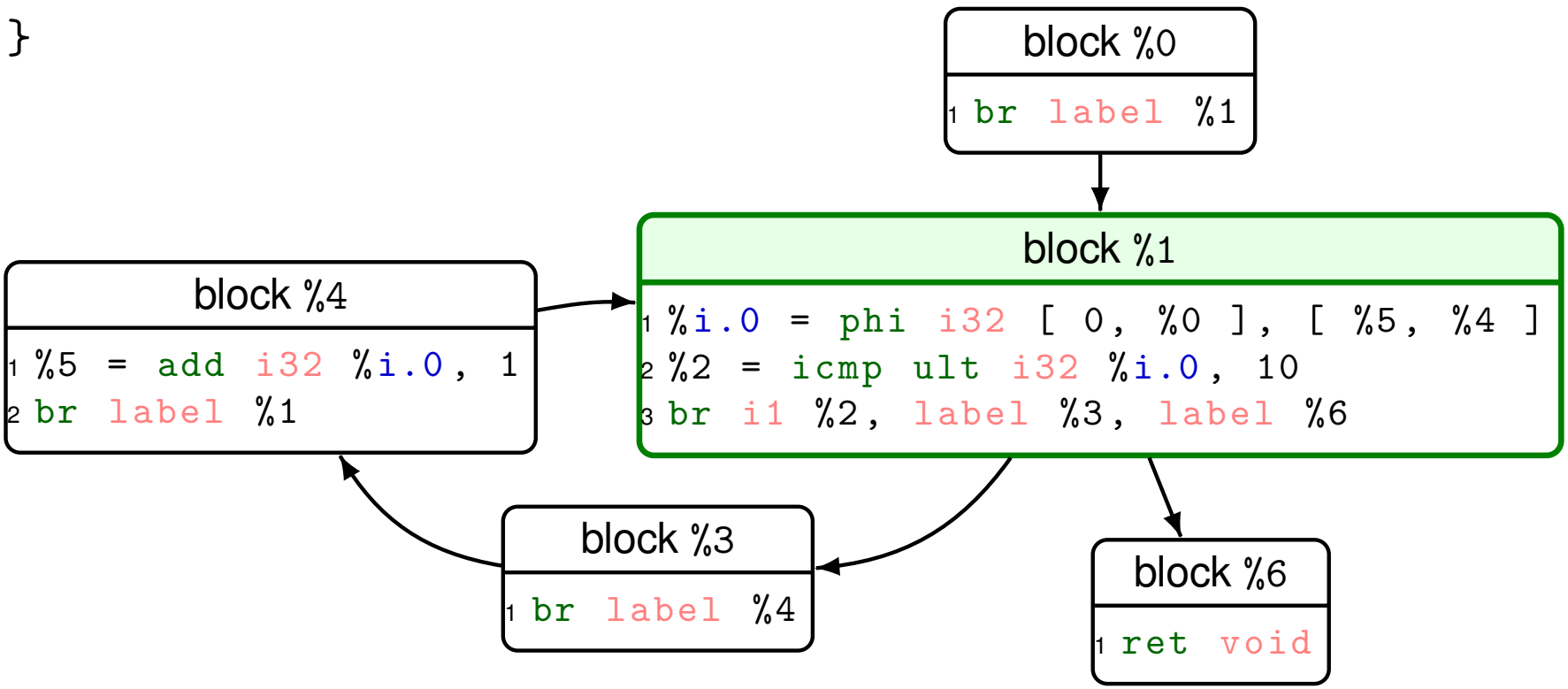
We encode the control points in the invariant, the transition, and store for u **vectors of vectors**.

In a nutshell

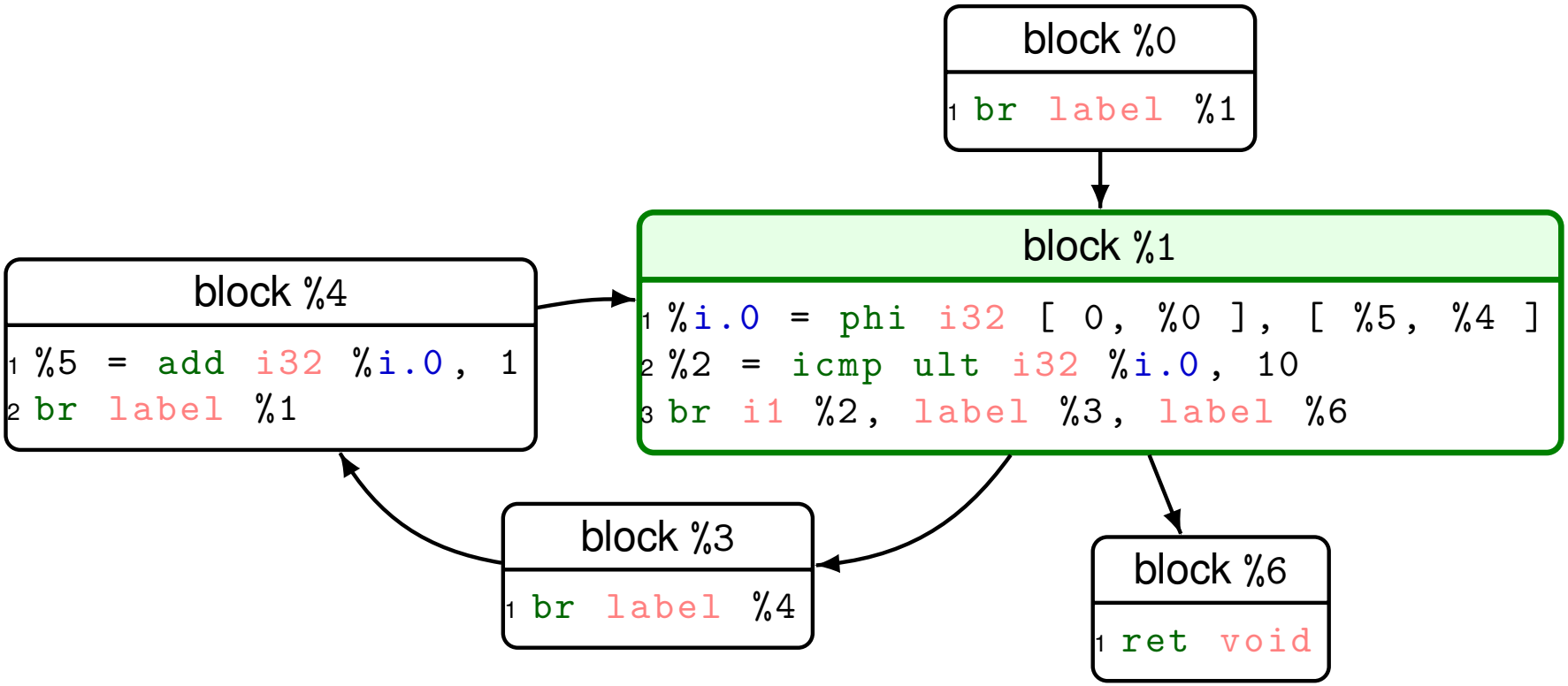


Control flow graph and LLVM representation

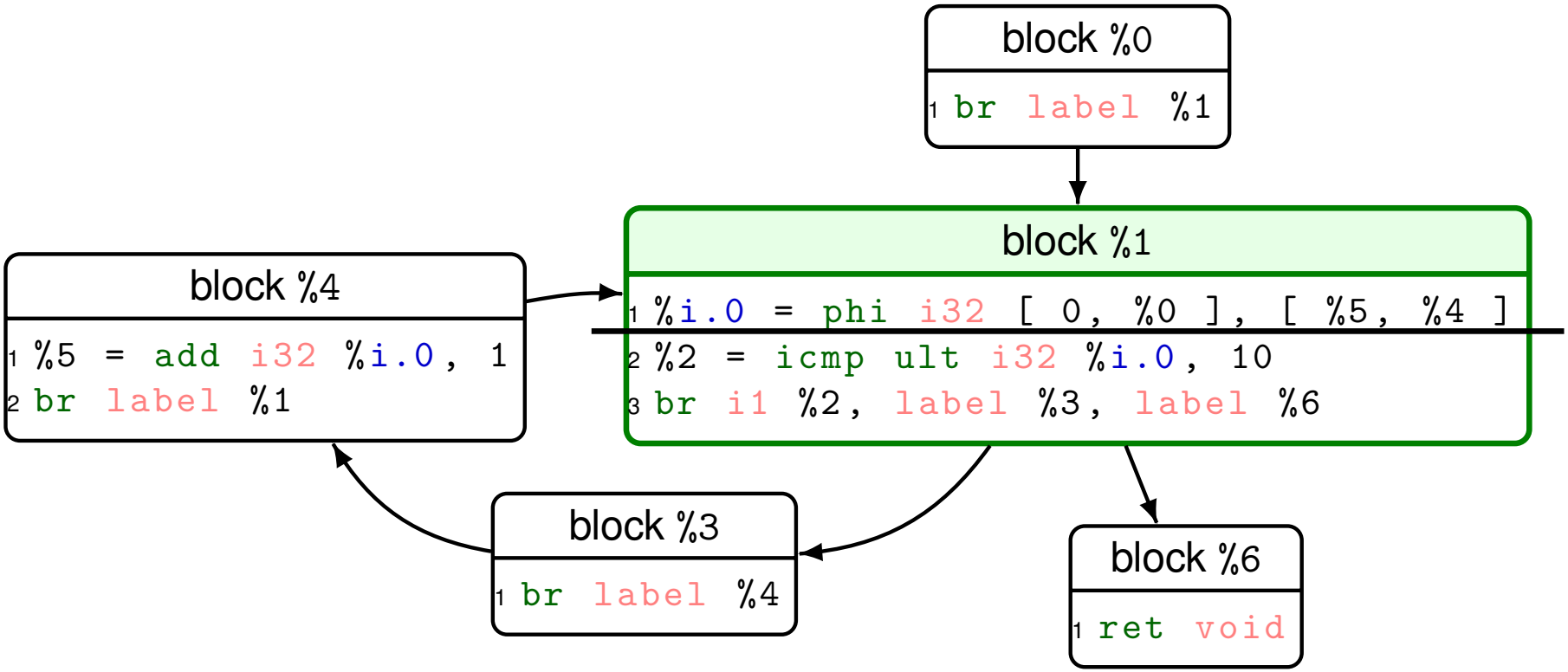
```
1 void simple_loop_constant() {
2   for(unsigned i=0; i<10; i++) {
3     // Do nothing
4   }
5 }
```



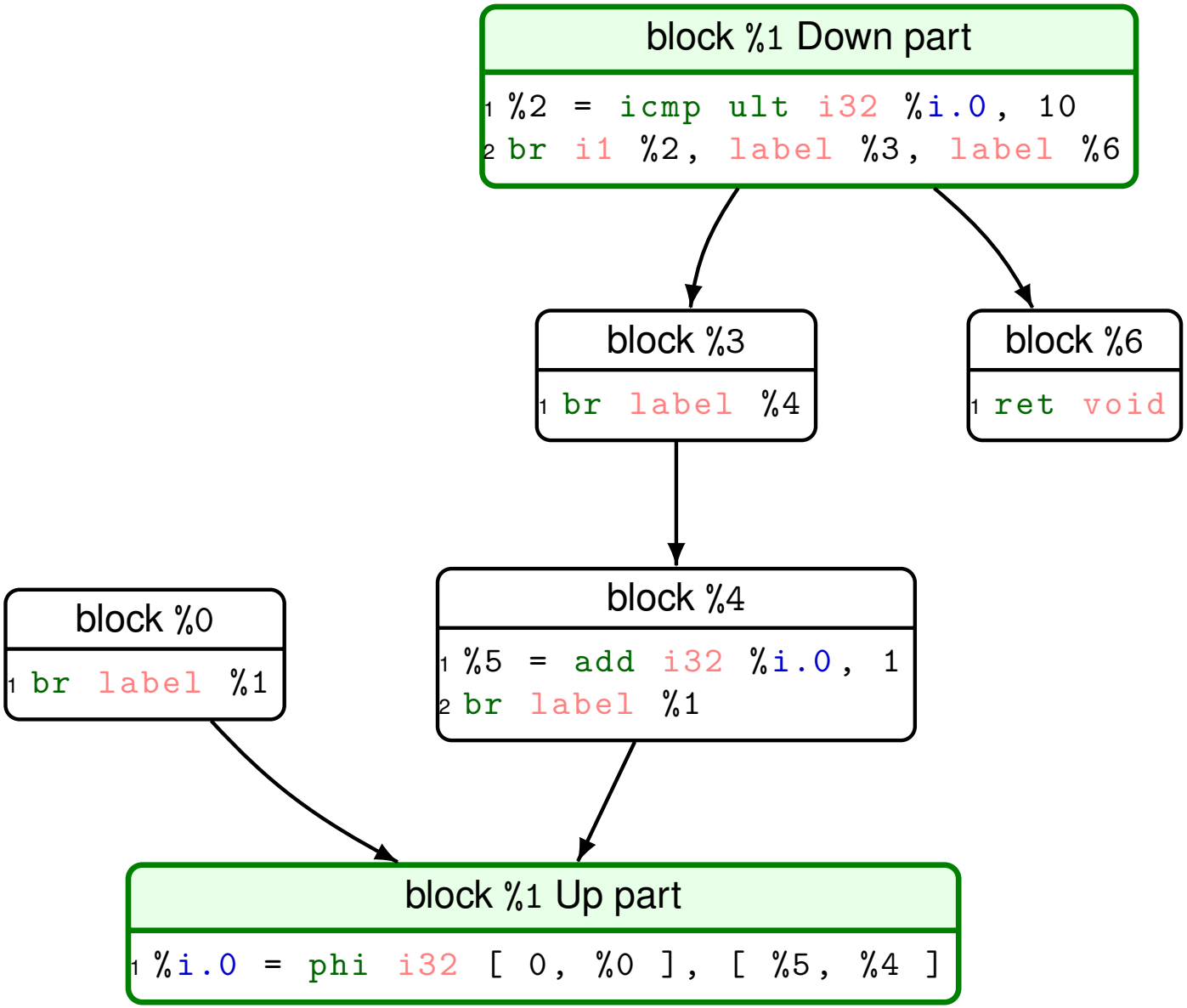
SMT encoding for control-flow-graph



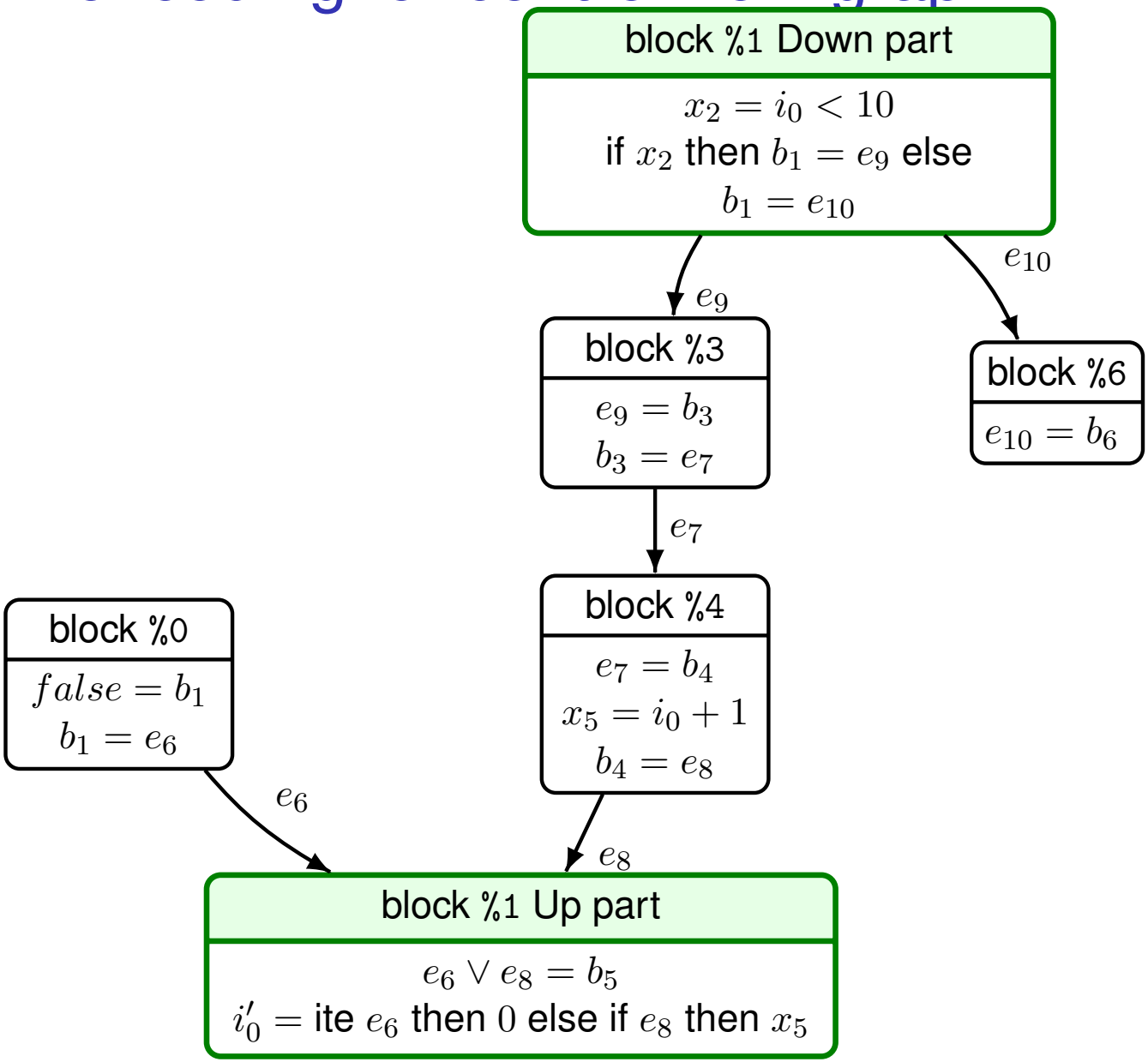
SMT encoding for control-flow-graph



SMT encoding for control-flow-graph



SMT encoding for control-flow-graph



Experiments

Example	Ranking function	Rank tool		Termite			
		#LP	Avg. #size	#LP	Avg. #size	#SMT	Avg. size
$The\ example$	$y+1$	1	84×51	1	3×7	3	20
$easy1$	$41-x$	1	334×155	1	3×6	3	21
$easy2$	z	2	86×42	1	3×5	3	16
$wcet2$	$-11i-j+65$	2	225×94	2	4×6	4	20
$exmini$	$102-i-j+k$	2	140×65	3	6×8	5	16
$cousot9$	$\binom{i}{j}$	3	180×75	5	6×8	6	25

Conclusion

Summary

We saw a method to infer ranking functions in an iterative fashion using extremal counter-examples

- which always terminate,
- which scales better than other approaches.

Future work

- consolidate Termite
- investiguate the complexities
- still room for improvement.

Transition system

Transition system

We consider programs over a state space $\mathcal{S} \subset \mathcal{W} \times \mathbb{Q}^n$, where :

- $\mathcal{W}$ is the finite set of control states, defined by an initial state and a transition relation τ ;
- $\mathbb{Q}^n$ is the value of the set of variable considered at the different control points.

Set of reachable values

We note

$$\mathcal{R}_k = \{\boldsymbol{x} \mid (k, \boldsymbol{x}) \in \mathcal{S}\}$$

the set of all values of $\boldsymbol{x}$ when the flow is in the state k.

Invariants

Invariant

An invariant on a control point $k \in \mathcal{W}$ is a formula $\phi_k(\boldsymbol{x})$ that is true for all reachable states $(k, \boldsymbol{x})$.

Affine invariant

An invariant is affine if it is a conjunction of a finite number of affine conditions on program variables.
Said in another way, for all $k \in \mathcal{W}$, there exists a convex polyhedron $\mathcal{P}_k$ such that $\mathcal{R}_k \subseteq \mathcal{P}_k$.

Linear ranking function

A (strict) linear ranking function
is a function $\rho : \mathcal{W} \times \mathbb{Q}^n \to \mathbb{Q}$ such that :

- for any state $k \in \mathcal{W}$, $\boldsymbol{x} \mapsto \rho(k, \boldsymbol{x})$ is affine linear ;
- for any transition $(k, \boldsymbol{x}, k', \boldsymbol{x}')$, $\rho(k', \boldsymbol{x}') \leqslant \rho(k, \boldsymbol{x}) - 1$;
- for any state $(k, \boldsymbol{x})$ in the invariant $\mathcal{I}$, $\rho(k, \boldsymbol{x}) \geqslant 0$;

Weak linear ranking function

We replaces the second condition by $\rho(k', \boldsymbol{x}') \leqslant \rho(k, \boldsymbol{x})$.

Lexicographic Linear ranking function

A **Lexicographic** (strict) linear ranking function **of dimension** m

is a function $\rho : \mathcal{W} \times \mathbb{Q}^n \to \mathbb{Q}^{\mathbf{m}}$ such that :

- for any state $k \in \mathcal{W}$, $\boldsymbol{x} \mapsto \rho(k, \boldsymbol{x})$ is affine linear ;
- for any transition $(k, \boldsymbol{x}, k', \boldsymbol{x}')$, $\rho(k', \boldsymbol{x}') \prec \rho(k, \boldsymbol{x})$;
- for any state $(k, \boldsymbol{x})$ in the invariant $\mathcal{I}$, **all coordinates of** $\rho(k, x)$ **are nonnegative** .

Weak **Lexicographic** linear ranking function

We replaces the second condition by $\rho(k', \boldsymbol{x}') \preceq \rho(k, \boldsymbol{x})$.

Lexicographic order

$\langle x_1, \ldots, x_m \rangle \prec \langle y_1, \ldots, y_m \rangle$ if and only if there exists an i such that $x_j = y_j$ for all $j < i$ and $x_i \leqslant y_i - 1$

Maximal termination power

Objective function

Given a (subset) $\mathcal{C}$ of generators of $\mathcal{P}_{\mathcal{I},\tau}$, $\rho = \boldsymbol{\lambda}\boldsymbol{x} + \ell$, we define $\pi_{\mathcal{C}}(\rho)$ the set of all elements u_i of $\mathcal{C}$ that satisfy $\boldsymbol{\lambda}.u_i > 0$.

Proposition

$\pi_{\mathcal{C}}(\rho)$ is max for cardinality iff max with respect to inclusion.

Final Algorithm for one control point

Require: $\mathcal{I}$ and τ
  $\mathcal{C} \leftarrow \emptyset$, $\mathcal{B} \leftarrow \emptyset$
  $finished \leftarrow$ false
  $\boldsymbol{\lambda} \leftarrow \mathbf{0}$, $\lambda_0 \leftarrow 0$
  while $\neg finished \wedge Sat(\mathcal{I} \wedge \tau \wedge AvoidSpace(\boldsymbol{u}, \mathcal{B}))$ with minimization for $\boldsymbol{\lambda} \cdot \boldsymbol{u} (\leqslant 0)$ **do**
    $(\boldsymbol{u}, unbound) \leftarrow$ a model for $\boldsymbol{u}$ in the above SMT test
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{\boldsymbol{u}\}$
    if $unbound$ **then**
      Let $\boldsymbol{r}$ a ray generator of $\mathcal{P}_H$ such that $\boldsymbol{u} = \cdots + \alpha \boldsymbol{r}$
      $\mathcal{C} \leftarrow \mathcal{C} \cup \{\boldsymbol{r}\}$
    end if
    $(\boldsymbol{\gamma}, \boldsymbol{\delta}) \leftarrow LP(\mathcal{C}, \text{Cons}_{\mathcal{I}})$
    if $\boldsymbol{\gamma} = \mathbf{0}$ **then** $finished \leftarrow$ true
    else
      $\boldsymbol{\lambda} \leftarrow \sum_{i=1}^{m} \gamma_i \boldsymbol{a}_i$, $\lambda_0 \leftarrow \sum_{i=1}^{m} \gamma_i b_i$
      if $\delta_{\boldsymbol{u}} = 0$ **then** $\mathcal{B} \leftarrow \mathcal{B} \cup \{\boldsymbol{u}\}$
      end if
    end if
  end while
  return $(\boldsymbol{\lambda}, \lambda_0, (\bigwedge_i \delta_i = 1) \wedge \neg Sat(\mathcal{I} \wedge \tau \wedge \boldsymbol{u} = \mathbf{0}))$

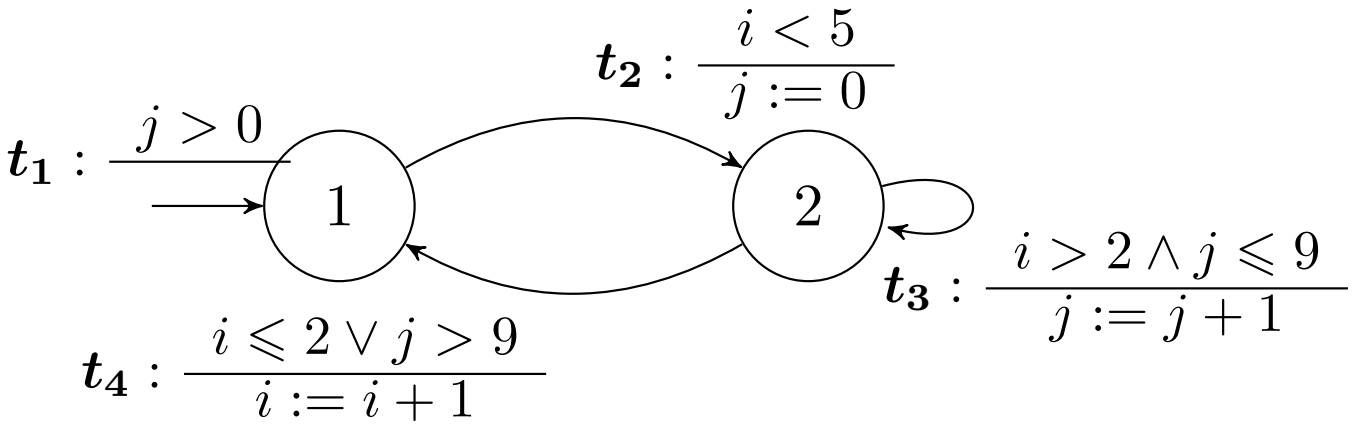
Multidim

Require: $\mathcal{I}$ and τ
  $d \leftarrow 1$, $failed \leftarrow$ false
  repeat
    $(\boldsymbol{\lambda}, \lambda_0, strict) \leftarrow$

$$\text{MONODIM}\left(\mathcal{I}, \tau \wedge \bigwedge_{d'<d} \boldsymbol{\lambda}_{d'} \cdot \boldsymbol{u} = 0\right)$$

    if $\neg strict$ **then**
      if $\boldsymbol{\lambda}$ is in the span of ρ **then**
        $failed \leftarrow$ true
      else
        $\rho_d \leftarrow \boldsymbol{\lambda} + \lambda_0$
        $d \leftarrow d + 1$
      end if
    end if
  until $strict \vee failed$
  return if $failed$ then “None” else ρ

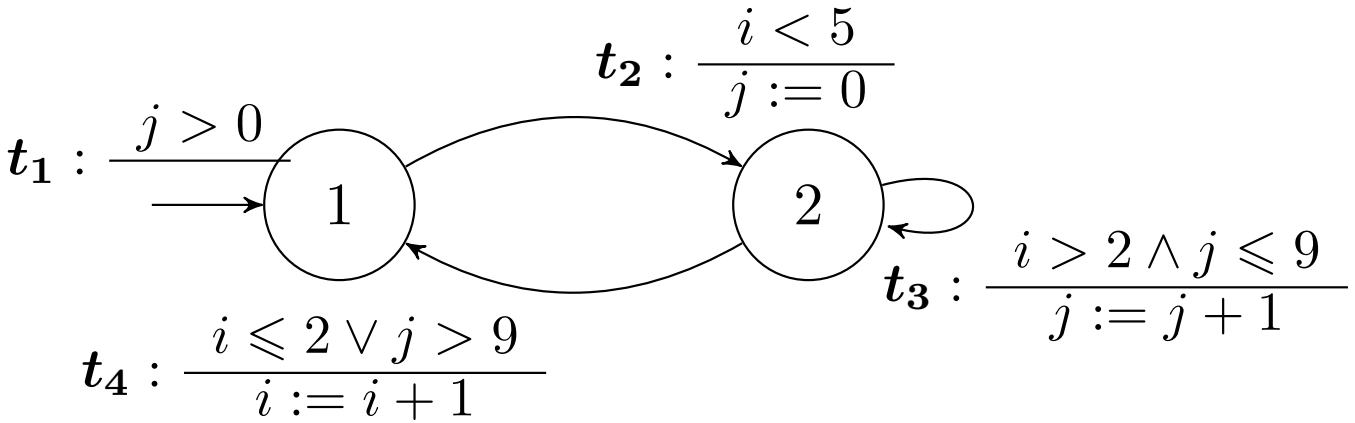
Multipc - example



1. Beginning with $\mathcal{C} = \{\}$ and $\rho(\boldsymbol{x}) = 0$, that is : $\boldsymbol{\lambda}^1 = \mathbf{0}$ and $\boldsymbol{\lambda}^2 = \mathbf{0}$. We have $\boldsymbol{x} = \binom{i}{j}$ and $\boldsymbol{u} = e^k(\boldsymbol{x}) - e^{k'}(\boldsymbol{x'})$. In the SMT-query, τ is now written as follows :

$$\left(k = 1 \wedge k' = 2 \implies i < 5 \wedge j' = 0 \wedge \boldsymbol{u} = (i, j, i', j')^\top\right) \wedge \ldots$$

Multipc - example



First iteration.

2. $Sat(\mathcal{I} \wedge \tau \wedge AvoidSpace(\boldsymbol{u}, \mathcal{B}) \wedge \mathbf{0} \cdot \boldsymbol{u} \leqslant 0)$?
Yes, with $k = 2$, $k' = 1$, $\boldsymbol{x} = \binom{1}{10}$ and $\boldsymbol{x}' = \binom{-2}{10}$ (this corresponds to transition $\mathbf{t_4}$)

3. $\mathcal{C} \leftarrow \left\{ \begin{pmatrix} 1 & 10 & -2 & -10 \end{pmatrix}^\top \right\}$

4. Call $LP(\mathcal{C}, \text{Cons}_{\mathcal{I}})$.
It gives us $\boldsymbol{\lambda}^1 = \binom{0}{0}$ and $\boldsymbol{\lambda}^2 = \binom{1/2}{0}$.

. . . few iterations

Return. We obtain $\rho^1(\boldsymbol{x}) = 0, \rho^2(i, j) = -11/2i - j + 32$, a strict ranking function for $(\tau, \mathcal{I})$